Instituto Superior de Economia e Gestão

UNIVERSIDADE TÉCNICA DE LISBOA

Curso de Mestrado
Gestão de Sistemas de Informação

2008 – 2009

Desenvolvimento de Modelos Baseados em Agentes

Plataforma Aplicacional

Dissertação de Mestrado

17-11-2010

Candidato: Paulo Boto

Orientação: Doutora Tanya Vianna de Araújo

Júri:

Prof. Doutor Mário Fernando Maciel Caldeira (em substituição da Prof. ª Doutora Maria Fernanda Abreu Sampaio) - Presidente

Prof.ª Doutora Tanya Vianna de Araújo - Vogal

Prof. Doutor Samuel Martins Eleutério - Vogal

Síntese

Apesar de a experimentação ser o método mais utilizado em muitas áreas científicas, nem sempre é possível recorrer a esse método nas áreas das Ciências Sociais ou das Ciências Económicas. Uma alternativa é a simulação computacional, em que se recorre a um programa de computador para representar um modelo do fenómeno a estudar, aplicando as abstracções e conceitos considerados relevantes e apropriados, tem-se vindo a tornar um instrumento cada vez mais utilizado e útil na pesquisa científica.

Ao longo dos últimos anos, um dos métodos de simulação computacional que tem tido maior desenvolvimento é o desenvolvimento de Modelos Baseado em Agentes: este consiste num sistema computacional que simula as acções das entidades intervenientes no fenómeno a estudar, e as interacções dessas entidades entre si e com o ambiente em que se encontram localizadas, tendo em vista a confirmação de hipóteses teóricas que contribuam para explicar o fenómeno estudado.

Em linhas gerais, o tema proposto para este Trabalho de Fim de Mestrado é a criação de uma plataforma aplicacional para a criação de Modelos Baseados em Agentes, contemplando os requisitos gerais deste tipo de simulação computacional, e avaliar as vantagens e desvantagens desta plataforma.

Esta dissertação começará por enquadrar teórica e historicamente a simulação computacional no âmbito das Ciências Sociais, para em seguida identificar os aspectos fundamentais e específicos dos Modelos Baseados em Agentes.

Sucessivamente, será concebida a arquitectura da plataforma aplicacional, considerando os requisitos gerias que se podem associar à criação de Modelos Baseados em Agentes e será efectuada a sua implementação; a plataforma será depois utilizada para a construção de vários

modelos, de modo a verificar a conveniência da sua utilização para construção de alguns dos modelos mais frequentes referidos na literatura.

Por fim, serão indicadas as várias possíveis evoluções e ampliações à plataforma criada, no sentido de a tornar mais completa, tanto sobre o ponto de vista das funcionalidades contempladas como do ponto de vista da sua versatilidade, e são avaliadas as vantagens e desvantagens da sua utilização.

Assim, em linhas gerais, a metodologia seguida no desenvolvimento deste trabalho será constituída pelas seguintes seis etapas principais:

1. Adquirir e consolidar o conhecimento sobre a modelização de fenómenos sociais baseada em agentes;

2. Especificar os principais requisitos que deverão ser contemplados pela plataforma e identificar a arquitectura mais apropriada;

3. Construir a plataforma;

4. Verificar a plataforma, utilizando-a para construir alguns Modelos Baseados em Agentes de referência descritos na literatura;

5. Identificar possíveis evoluções e extensões tendo em vista obter funcionalidade adicional e uma maior versatilidade da plataforma desenvolvida;

6. Avaliar as vantagens e desvantagens da utilização da plataforma desenvolvida.

Palavras-chave

Modelos Baseados em Agentes, simulação computacional, sociedades artificiais, Economia Computacional

Abstract

Despite the experimental method being the most widely used method in many scientific areas, is not always possible to use this method in the areas of Social Sciences or Economics Sciences. An alternative is the computer simulation, where a computer program is used to represent a model of the phenomenon to study, applying the abstractions and concepts considered relevant and appropriate, has been used as a key instrument in research.

During the last years, one of the methods of computer simulation which has had significant development is the so called Agent-Based Models method: this method is based on computational systems that simulate the actions of the entities involved in the phenomena to study, and the interactions of these entities among themselves and with the environments where they are located, in order to explain the phenomenon studied.

The theme proposed for this master thesis is to create an application platform for creating agent-based models covering the general requirements of this type of computer simulation, and to evaluate advantages and drawbacks in its use.

The thesis will begin by describing, historically and theoretically, computer simulation as applied to social sciences, to then identify the basic and specific aspects of Agent-Based Models.

Then, the application platform architecture will be designed, considering the general requirements that may involved in the creation of Agent-Based Models and its implementation will be carried out; finally, the platform will be used to build several models in order to verify the appropriateness of their use for building some of the models most frequently mentioned in literature.

At the end, the various possible developments and extensions to the platform in order to make it more complete (considering functionality and versatility) will be described and the advantages and drawbacks of its use will be evaluated as well.

Thus, in general, the methodology applied in developing this master thesis will pursue the subsequent six main steps:

1. Acquire and consolidate knowledge on social phenomena models based on agents;
2. Identify key issues to be addressed by the platform and identify the most appropriate architecture options;
3. Design and implement the platform;
4. Check the platform, using it to build some agent-based models described in the literature;
5. Identify possible developments and extensions in order to obtain additional functionality and versatility;
6. Evaluate the platform advantages and drawbacks.

Keywords

Agent-Based Models, computational simulation, artificial societies, Computational Economics

Índice Geral

Índice das Figuras

Índice das Tabelas

Agradecimentos

Em primeiro lugar agradeço a Profª. Doutora Tanya Vianna de Araújo cuja orientação foi determinante para a identificação do tema deste Trabalho de Fim de Mestrado e cuja permanente disponibilidade para esclarecer todas as questões que surgiram no decurso do trabalho desenvolvido foi crucial para o resultado obtido.

Depois desejo agradecer os corpos docentes do curso de Pós-Graduação em Sistemas e Tecnologias de Informação para as Organizações (ano lectivo 2007/2008) e do curso de mestrado de Gestão de Sistemas de Informação (ano lectivo 2008/2009), do Instituto Superior de Economia e Gestão, que me proporcionaram a possibilidade de adquirir e consolidar vários conceitos, métodos e técnicas utilizados no desenvolvimento do presente Trabalho de Fim de Mestrado.

Por fim, não posso deixar de agradecer os meus colegas de grupo (João Almeida, João Morais, Norberto Correia e Ricardo Pires) do referido curso de pós-graduação, cujo incentivo foi determinante para a minha decisão de realizar o curso de mestrado.

1. Contexto e Objectivo do Trabalho de Fim de Mestrado

1.1. Contexto

Na área das Ciências Sociais ou Económicas não é com muita frequência possível recorrer à experimentação para a confirmação de hipóteses teóricas elaboradas para a explicação dos fenómenos estudados. Esta limitação é provocada por motivos de ordem prática (muitos fenómenos não são simplesmente replicáveis) ou por motivos de ordem moral (não é possível efectuar experiências que possam causar directa ou indirectamente prejuízo às pessoas envolvidas no fenómeno estudado), entre outros.

Assim, e tirando partido do enorme progresso verificado, ao longo das últimas décadas, na sofisticação, acessibilidade e difusão das tecnologias de informação, o estudo dos fenómenos sociais e económicos tem vindo a utilizar cada vez mais estas tecnologias para efectuar simulações dos fenómenos estudados.

Dada a extrema relevância que, no tipo de fenómenos em questão, os agentes sociais ou económicos representam, é natural que uma das técnicas mais comuns de modelização seja a que utiliza as chamadas Abordagens Baseadas em Agentes ou Modelos Baseados em Agentes. Este tipo de modelização, recorrendo às várias possibilidades oferecidas pelos sistemas informáticos, tanto ao nível do *hardware* como do *software* disponíveis, é muito adequado à representação do comportamento de entidades, com características de percepção, de tomada de decisão e de tomada de acção, típicas dos agentes sociais ou económicos no âmbito dos problemas estudados.

1.2. Tema e Questão Central de Investigação

O tema desenvolvido pelo presente Trabalho de Fim de Mestrado é a criação de uma plataforma aplicacional para a construção de Modelos Baseados em Agentes.

Pretende-se que a plataforma criada seja genérica, cumprindo um Quadro de Referência Conceptual, o qual será também definido no âmbito do trabalho, de modo a permitir a construção de vários tipos de modelos que tenham efectivamente uma aplicação prática e fiável ao estudo de fenómenos de vários tipos, fundamentalmente nas áreas das Ciências Sociais e das Ciências Económicas.

Para orientar o desenvolvimento do tema escolhido, assumir-se-á como questão central de investigação a seguinte:

Quais as vantagens e quais as desvantagens de uma plataforma aplicacional genérica para o desenvolvimento de Modelos Baseados em Agentes?

1.3. Objectivos do Trabalho

Considerando o tema escolhido e a questão central de investigação apresentada, os quatro objectivos seguintes orientarão o presente trabalho:

- Criar uma plataforma aplicacional para o desenvolvimento de Modelos Baseados em Agentes;
- Descrever exemplos práticos de aplicação desta plataforma considerando alguns modelos de referência disponíveis na literatura;
- Apresentar extensões e evoluções à plataforma criada;

- E, por fim, responder à questão de investigação referindo quais as vantagens e desvantagens da utilização da plataforma desenvolvida.

1.4. Motivação

A principal motivação para o tema escolhido foi a aquisição de conhecimento, por parte do candidato, na área dos Modelos Baseados em Agentes, procurando tirar partido da experiência e conhecimento que este possui na área do desenvolvimento de *Software*, resultantes da sua actividade profissional: gestão de projectos de desenvolvimento de sistemas informáticos.

Por outro lado, outra motivação que orientou a escolha do tema foi criar uma oportunidade de consolidação e aprofundamento das técnicas de programação orientada por objectos, de forma a actualizar os conhecimentos do candidato nesta área.

Ou seja, em linhas gerais, a escolha do tema do presente Trabalho de Fim de Mestrado foi feita de modo a conseguir obter, não só, um resultado útil e apropriado do ponto de vista académico mas também um resultado útil e apropriado para a actividade profissional do candidato.

Convém realçar que a escolha do tema resultou, em última instância, na disponibilização de uma plataforma aplicacional que certamente poderá ser aplicada na criação de Modelos Baseados em Agentes no âmbito de trabalhos concretos de investigação em várias áreas científicas apesar do contexto considerado ter sido sempre o das Ciências Sociais e das Ciências Económicas.

1.5. Metodologia

A plataforma aplicacional desenvolvida foi criada no âmbito de um quadro de referência conceptual fundamentado na teoria subjacente aos Modelos Baseados em Agentes de acordo com os elementos recolhidos na pesquisa bibliográfica realizada. Procurou-se, assim, por um lado, criar uma plataforma o mais possível genérica, mas sempre aderente ao quadro de referência conceptual, e, por outro lado, obter uma consolidação dos conhecimentos adquiridos.

A arquitectura e o ambiente de programação foram escolhidos de forma a permitir um aprofundamento dos conceitos relacionados com a programação orientada por objectos, tendo em vista a necessidade de actualização de conhecimentos nesta área por parte do candidato.

A metodologia que foi seguida no desenvolvimento de todo o trabalho foi constituída por cinco etapas principais enquadradas em duas fases.

A primeira fase contemplou a recolha e consolidação de informação, bem como a definição dos requisitos relevantes a considerar no desenvolvimento da plataforma. Esta fase, assim, compreendeu as seguintes etapas:

1. Aquisição e consolidação do conhecimento, contemplando os conceitos e técnicas relacionados com a modelização de fenómenos sociais baseada em agentes;

2. Especificação dos requisitos que condicionam a arquitectura, em geral, e o desenho da plataforma, em particular.

A segunda fase abrangeu a construção da plataforma e a sua verificação, e terminou com a avaliação da plataforma no sentido de identificar melhoramentos e extensões. Esta fase, assim, compreendeu as seguintes etapas:

3. Desenho e codificação da plataforma;

4. Verificação da plataforma, aplicando-a na criação de alguns Modelos Baseados em Agentes que se podem considerar de referência na área das Ciências Sociais;

5. Identificação das evoluções e extensões possíveis, tendo em vista tornar a plataforma mais completa ao nível funcional e adequada a mais tipos de modelos.

1.6. Estrutura da Dissertação

O presente documento constitui a dissertação resultante do Trabalho de Fim de Mestrado, sendo constituído por seis capítulos:

- O capítulo 1 ("Contexto e Objectivo do Trabalho de Fim de Mestrado"), o actual, descreve o tema da dissertação e a Questão Central de Investigação, os objectivos e motivações que orientaram a sua escolha e a metodologia seguida no desenvolvimento do trabalho;
- O capítulo 2 ("Enquadramento Geral e Perspectiva Histórica") enquadra a modelização baseada em agentes, sobretudo numa perspectiva histórica e teórica, relacionando-a com o tema mais alargado da simulação computacional;
- O capítulo 3 ("Modelos Baseados em Agentes") descreve os conceitos e aspectos relevantes relacionados com Modelos Baseados em Agentes;
- O capítulo 4 ("Quadro de Referência Conceptual") apresenta o quadro de referência que servirá de orientação ao desenho da plataforma aplicacional;
- O capítulo 5 ("A Plataforma Tecnológica") descreve os aspectos principais relacionados com as tecnologias utilizadas na criação da plataforma aplicacional;

- O capítulo 6 ("A Plataforma Aplicacional") apresenta detalhadamente a arquitectura da plataforma aplicacional desenvolvida;
- O capítulo 7 ("Exemplos de Aplicação") descreve a aplicação da plataforma à construção de quatro modelos concretos que se podem considerar referências na área das Ciências Sociais;
- O capítulo 8 ("Evolução da Plataforma Aplicacional"), depois de um resumo dos objectivos e resultados obtidos, apresenta uma análise crítica ao trabalho realizado, indica as principais evoluções e extensões à plataforma desenvolvida;
- O capítulo 9 ("Conclusão") responde à Questão Central de Investigação, expondo as vantagens e desvantagens da utilização da plataforma aplicacional criada, e compara os resultados obtidos com os objectivos do trabalho;
- Por fim, o capítulo 10 ("Bibliografia") indica todas as fontes bibliográficas referidas no texto.

2. Enquadramento Geral e Perspectiva Histórica

2.1. Sistemas Complexos

A impossibilidade de prever os futuros estados de um sistema, a partir do conhecimento que se tem dos elementos que o constituem, é um dos critérios utilizados para classificar um sistema como complexo; outro critério é a emergência de estruturas ou padrões como resultado da evolução dinâmica do sistema. O primeiro critério é utilizado nos sistemas com poucos graus de liberdade que, apesar de serem simples, poderão ter comportamento complexo; o segundo critério é utilizado nos sistemas com muitos graus de liberdade (constituídos por muitos elementos) que apresentam um comportamento colectivo simples (Araújo, 2010).

Os fenómenos sociais, estudados no âmbito das Ciências Sociais e de outras áreas científicas, como as Ciências Económicas por exemplo, contemplam geralmente sistemas que apresentam comportamento complexo, na perspectiva do segundo critério referido acima: sistemas com muitos graus de liberdade com emergência de padrões colectivos simples. De facto, nestes fenómenos participam várias entidades (pessoas, organizações, instituições, empresas) que se relacionam entre si, resultando trocas de informação (ou, mais em geral, de valor) que se reflectem no futuro comportamento das próprias entidades: não é portanto possível prever a evolução destes sistemas, simplesmente com base no estudo isolado dos elementos que os constituem sem considerar a interacção entre estes elementos. Os atributos e estruturas globais dos sistemas sociais emergem assim como resultado das acções das entidades elementares (por sua vez condicionadas pelas interacções entre essas entidades), utilizando-se neste contexto o termo “emergente” para indicar que as propriedades utilizadas na

caracterização dos atributos e estruturas são propriedades colectivas (e espontâneas) do sistema (Gilbert, 2004).

Outra área em que o estudo dos sistemas complexos tem vindo a ter um impacto cada vez maior é a área das Ciências Económicas, onde se tem vindo a desenvolver e consolidar um ramo denominado Economia Computacional Baseada em Agentes (*Agent-Based Computational Economics*, ACE). Nesta área de pesquisa, economias de mercado descentralizadas são estudadas como sistemas complexos constituídos por grande número de agentes autónomos, interagindo entre si e com o ambiente, e com capacidade de adaptação às mudanças percepcionadas. Estas interacções, ao longo do tempo, dão origem a padrões macroeconómicos que, por sua vez, influenciam o comportamento dos agentes (Tesfatsion, 2002).

Através de instrumentos de modelização disponíveis na área da Economia Computacional Baseada em Agentes, os investigadores conseguem hoje estudar o funcionamento de economias descentralizadas considerando aspectos diversos como, por exemplo, vários mecanismos de aprendizagem nos agentes económicos, o paradigma da concorrência imperfeita, a formação de redes comerciais e a evolução da relação entre os comportamentos individuais e as organizações económicas.

2.2. A Simulação como Instrumento de Investigação

A utilização da simulação, como técnica de modelização de sistemas sociais, tem vindo a crescer significativamente no estudo das Ciências Sociais. A técnica da simulação pode ser utilizada (Axelrod 2005):

- Para prever resultados a partir de vários valores de entrada, eventualmente variáveis no tempo, e aplicando mecanismos que representam as hipóteses teóricas consideradas;
- Na execução de tarefas no domínio da inteligência artificial (por exemplo diagnóstico médicos e reconhecimento verbal);
- No treino de utilizadores em ambiente interactivo possibilitando uma reprodução da realidade adequada (por exemplo pilotagem de aviões);
- No ensino, através da criação de cenários artificiais que permitem percepcionar relações entre as entidades estudadas e identificar princípios teóricos;
- Para exemplificar hipóteses teóricas (por exemplo no jogo da vida de Conway em que se demonstra que sistemas complexos podem ser criados a partir de regras muito simples);
- Na pesquisa científica, através da demonstração de hipóteses e como meio que permite a descoberta de novas hipóteses teóricas.

A utilização da técnica da simulação computacional como método científico surge como alternativa ou complemento dos métodos científicos clássicos:

- O método indutivo em que se extraem hipóteses teóricas ou padrões a partir da análise dos dados recolhidos;
- O método dedutivo em que se obtêm conclusões teóricas a partir de princípios e axiomas assumidos.

2.3. A Evolução da Simulação Computacional nas Ciências Sociais

Apesar de a experimentação ser o método mais utilizado em muitas áreas científicas, é geralmente impossível recorrer a esse método nas Ciências Sociais. A experimentação comporta alterar um sistema isolado e observar as consequências dessa alteração, comparando esse sistema com outro igual mantido inalterado, de modo a assegurar que os resultados observados são provocados pela alteração efectuada. Nos sistemas sociais, este método de investigação quase sempre não é possível, por motivos éticos e práticos (Gilbert, 2007).

Assim, a investigação científica, na área das Ciências Sociais, comporta a criação de uma representação simplificada do fenómeno social a estudar, sendo dado o nome de "modelo" a esta representação simplificada. O modelo poderá ser puramente verbal (como, por exemplo, na investigação histórica) ou poderá ser formalmente expresso por equações matemáticas (como, por exemplo, no estudo de teorias económicas).

A principal desvantagem dos modelos matemáticos para o estudo dos fenómenos sociais prende-se com a complexidade desses modelos: a solução analítica das equações constituintes do modelo matemático é geralmente impossível em virtude destas equações terem que descrever relações, geralmente não lineares, entre as várias propriedades do fenómeno. Por outro lado, simplificações introduzidas para tornar o modelo matemático solúvel poderão comprometer a qualidade do modelo e dar origem a conclusões teóricas erradas (esta situação encontra-se frequentemente no estudo da Economia em que se consideram muitas vezes hipóteses de "total racionalidade" ou de "total informação" de modo a tornar resolúveis os modelos analíticos propostos) (Gilbert et al., 2000).

Para além destes dois tipos de modelos, modelo verbal e modelo matemático, existe um terceiro tipo de modelização: a simulação computacional. Na simulação computacional

recorre-se a um programa de computador para representar um modelo do fenómeno social, utilizando as abstracções e conceitos considerados relevantes e apropriados ao fenómeno que se pretende estudar e, se necessário, modelizando relações não lineares entre as propriedades do fenómeno. Assim, através desse programa, procura-se confirmar hipóteses de investigação, comparando os resultados obtidos na simulação a partir de observações de casos concretos (Gilbert et al., 2000).

Ao longo dos últimos cinquenta anos, podem-se considerar fundamentalmente três métodos utilizados na simulação computacional (Gilbert et al., 2005):

- Análise e Reconstrução de Sistemas Dinâmicos;
- Microsimulação;
- Desenvolvimento de Modelos Baseados em Agentes.

Apesar destes três métodos terem sido criados em épocas diferentes e poder-se dar uma perspectiva evolutiva à forma como cada método surgiu, são todos eles actualmente utilizados no estudo dos fenómenos sociais, tendo cada método vantagens específicas, o que leva a que a escolha do método deverá depender do tipo de fenómeno a estudar.

Nos anos 60 do século passado, começou-se a aplicar computadores para resolver conjuntos de equações diferenciais (ou às diferenças) representando a evolução, ao longo do tempo, de propriedades, globais e interligadas, relativas a populações abrangidas por um dado fenómeno social. Este tipo de modelização tem o nome de "Análise e Reconstrução de Sistemas Dinâmicos" e é caracterizado pelo facto que o estudo do fenómeno social não ser feito através da análise do comportamento dos vários agentes intervenientes e das suas interacções, mas, pelo contrário, ser feito através da modelização das variáveis globais que caracterizam o fenómeno (Gilbert, 2007). Assim, este método é sobretudo indicado para o estudo de fenómenos que abranjam uma população constituída por agentes homogéneos, com um

comportamento muito semelhante entre si, em que não seja relevante a heterogeneidade entre tipos de agentes. O método também não é recomendado para as situações em que seja relevante a aprendizagem dos agentes com base na experiência efectuada, em virtude da dificuldade em modelizar matematicamente estes aspectos.

A partir dos anos 70, surgiu o método denominado "Microsimulação". Neste método, a modelização do fenómeno social assenta sobre uma base de dados contendo toda a informação relevante, obtida por exemplo por meio de inquéritos, relativa a uma amostra suficientemente representativa das entidades (indivíduos, grupos ou organizações) intervenientes no fenómeno a estudar. O comportamento do modelo ao longo do tempo é definido matematicamente através de um conjunto de equações que representam as regras de evolução do modelo. Dado que a evolução do modelo não é determinística, estas equações terão que contemplar as probabilidades empíricas na determinação dos valores futuros das propriedades.

Por exemplo, pode-se utilizar a Microsimulação para estudar, numa perspectiva social, a evolução da população de um país, com base no censo realizado num dado momento, considerando propriedades de cada indivíduo como a idade, o sexo, o nível de escolaridade, o rendimento, a situação profissional, etc. Após se definir, matematicamente e numa perspectiva probabilística, como estas propriedades evoluirão ao longo dos anos, será possível, determinar valores para propriedades globais da população considerada, como, por exemplo, o rendimento médio, a percentagem de indivíduos reformados, etc. Ou seja, utilizando o modelo criado, poder-se-ão obter previsões estatísticas da população agregando a evolução prevista das várias propriedades dos indivíduos constituintes essa população (Gilbert, 2007). Entende-se, assim, a razão de ser do nome deste método, uma vez que o mesmo pressupõe simular a evolução das propriedades dos indivíduos (visão micro) em contraposição às propriedades globais da população (visão macro). O método da Microsimulação tem sido

utilizado para a previsão das consequências de alterações nas políticas relativas à cobrança de impostos e à segurança social, por exemplo.

O facto de o método da Microsimulação assentar sobre uma amostra real da população sujeita ao fenómeno social que se pretende estudar, e sendo a amostra obtida através de inquéritos efectivos (em contraposição a um conjunto de agentes criados aleatoriamente) consiste numa das principais vantagens do método. É relativamente fácil obter previsões fiáveis do estado de uma população após um determinado período de tempo (obviamente que a fiabilidade desta previsão dependerá de quão correctas estão as equações criadas para definir o modelo). Como desvantagens significativas deste método, podem-se referir essencialmente dois aspectos: um tem a ver com a dificuldade em modelar a evolução probabilística da população (ou seja, encontrar os parâmetros certos que descrevam como a população irá evoluir ao longo do tempo) e o outro está relacionado com o facto de os agentes neste tipo de modelização não interagirem entre si e de não existir o conceito de localização espacial de cada agente (Gilbert, 2007).

A terceira fase da simulação social surgiu com os Modelos Baseados em Agentes, nos anos 80, quando os primeiros computadores pessoais começaram a ser divulgados. A abordagem é, neste caso, também do tipo "*bottom up*", em que o modelo é criado a partir dos elementos intervenientes no fenómeno, só que agora, esses elementos (agentes) interagem entre si, sem nenhuma orquestração centralizada, seguindo regras simples e, muitas vezes, adaptando o próprio comportamento com base nos resultados obtidos. Assim, é criada uma representação simplificada do fenómeno social mas que tem a capacidade de evidenciar os aspectos relevantes da realidade social (Gilbert, 2007).

Os Modelos Baseados em Agentes têm vindo a ser cada vez mais utilizados no estudo dos fenómenos sociais porque permitem contemplar os comportamentos heterogéneos dos agentes envolvidos nos fenómenos sociais, comportamentos esses derivantes dos vários critérios de

decisão que os próprios agentes utilizam face ao ambiente em que se encontram inseridos e tendo em conta os objectivos pretendidos. Por outro lado, os Modelos Baseados em Agentes permitem também analisar estruturas emergentes da acção individual de cada agente, contemplando mecanismos de adaptação e aprendizagem específicos (Gilbert, 2007).

Em conclusão, os processos sociais compreendem entidades que se relacionam entre si de forma complexa, não linear e nalguns casos gerando estruturas auto-organizadas. O estudo destes processos é mais fácil e eficaz se, em vez de se tentar modelizar o processo como um todo, de forma global, se modelizarem as características relevantes das entidades, no que respeita às interacções entre si e com o ambiente (Macy et al., 2001).

2.4. Laboratórios computacionais

No contexto considerado na presente dissertação, deve-se entender um laboratório computacional como um ambiente artificial que permite efectuar experiências controladas e replicáveis de fenómenos sociais e económicos envolvendo agentes autónomos interagindo entre si e com o ambiente (Tesfatsion, 2002).

Inúmeras questões se levantam relacionadas com a construção de laboratórios computacionais tendo em vista a realização de experiências que se demonstrem eficazes, no sentido de permitir obter resultados relevantes para o progresso científico, e eficientes, no sentido de permitir obter esses resultados com recursos limitados, ao nível dos meios e do tempo envolvidos. Algumas dessas questões são:

- Plataforma

  Será melhor investir numa plataforma genérica e multifacetada, comum a vários tipos de simulações, aplicada transversalmente em vários campos da investigação

científica? Ou alternativamente será melhor ter um ambiente específico construído à medida das necessidades de uma determinada simulação?

- Apresentação dos resultados

  Quais são os mecanismos mais eficientes de apresentação dos resultados das simulações efectuadas de modo a não sobrecarregar os destinatários com informação redundante e desnecessárias?

- Validação externa

  Como se poderão validar adequadamente os resultados obtidos considerando os resultados observados através de outros meios?

- Verificação interna

  Até que ponto se pode assegurar que os resultados obtidos numa simulação realizada com um laboratório computacional não são influenciados, ou até provocados, por características, ou até defeitos, nos instrumentos utilizados no laboratório, como por exemplo erros nos programas desenvolvidos?

2.5. Modelos Baseados em Agentes no estudo de sistemas complexos

É muito fácil e intuitivo olhar para um bando de aves a voar e identificar esse conjunto de aves como uma única entidade com propriedades próprias (por exemplo, a forma) e com um objectivo. No entanto, não existe um organizador dessa entidade que defina a coreografia do conjunto: ela resulta somente das interacções entre as aves que constituem o bando ou seja cada ave reage ao movimento das aves adjacentes e vice-versa (Macy et al., 2001).

Pode-se criar com bastante facilidade um modelo que replique o comportamento do bando de aves, modelizando a relação entre as aves com base em três regras simples:

- Regra da separação (cada ave não se poderá aproximar a outra ave ou a um objecto qualquer mais do que uma distância mínima pré-fixada)
- Regra do alinhamento (cada ave deve procurar permanentemente estar alinhada com as aves adjacentes)
- Regra da coesão (cada ave deve permanentemente dirigir-se para a centro de massa do grupo das aves adjacentes)

Craig Reynolds mostrou, em 1987, como um modelo cujos elementos cumpram esta três simples regras, reproduz muito fielmente o comportamento de um bando de aves. Seria muito mais difícil, e provavelmente impossível com os meios actualmente disponíveis, obter uma simulação do comportamento global do bando de aves procurando descrever matematicamente esse comportamento com base nas propriedades observáveis ao nível de cada ave. No entanto, este é um bom exemplo, em que se consegue facilmente reproduzir uma organização e padrões de comportamento do sistema global modelizando as relações existentes entre os seus elementos.

Este é o princípio sobre o qual se baseia um dos principais instrumentos actualmente utilizados em várias áreas científicas: a Modelação Baseada em Agentes.

Em particular, e analogamente ao bando de aves, na área das Ciências Sociais, em que se estudam grupos sociais cujos elementos se relacionam entre si, de forma complexa e não linear, a identificação de propriedades e padrões emergentes é possível, e geralmente muito mais facilmente do que utilizando abordagens analíticas, a partir da modelização das relações entre os elementos do grupo (Macy et al., 2001).

3. Modelos Baseados em Agentes

Os Modelos Baseados em Agentes são um instrumento adequado para o estudo de sistemas que tenham as seguintes duas propriedades (Axelrod et al., 2005):

- Sejam constituídos por entidades que interagem entre si e com o ambiente em que se encontram;
- Apresentem estruturas emergentes das interacções entre as entidades que constituem o sistema, as quais não podem ser derivadas da agregação das propriedades das entidades individuais.

Quando o comportamento das entidades constituintes do sistema depende das interacções ocorridas entre essas entidades, e também com o ambiente, é muito difícil obter, através de métodos matemáticos analíticos e dedutivos, uma previsão sobre os futuros estados do sistema (resultantes da evolução dinâmica do sistema). Nesta situação a modelização baseada em agentes permite chegar a resultados passíveis de interpretação teórica e de aplicação prática.

3.1. Conceitos Básicos

Um Modelo Baseado em Agentes é um sistema computacional que simula as acções das entidades intervenientes no fenómeno social, e que simula as interacções dessas entidades entre si e com o ambiente, tendo em vista a confirmação de hipóteses teóricas elaboradas sobre um determinado fenómeno (Axtell, 2000).

Segundo a referência (Macal et al., 2005), um agente é uma entidade capaz de tomar acções independentes e autónomas e de reagir ao ambiente em que se encontra de modo a atingir um objectivo pré determinado.

A partir desta definição de agente, realçam-se as seguintes características como as mais significativas (Gilbert 2004):

- Um agente é uma entidade identificável com atributos próprios (propriedades e regras de comportamento e de tomada de decisão);
- Um agente participa num ambiente onde interage com outros Agentes e com o próprio ambiente;
- Um agente pode ser orientado por um objectivo, agindo autonomamente no sentido de atingir esses objectivo;
- Um agente pode adaptar, ao longo do tempo, o seu comportamento com base na experiência adquirida.

De forma mais concisa e precisa, um agente pode ser caracterizado pelas seguintes propriedades (Gilbert et al., 2000):

- Autonomia;
- Sociabilidade;
- Adaptabilidade;
- Intencionalidade.

Autonomia significa essencialmente capacidade de escolher, por si próprio, a acção a tomar.

Sociabilidade é a capacidade de interagir com os outros agentes através de algum mecanismo definido no próprio modelo.

Adaptabilidade (ou reactividade) é a propriedade que permite o agente percepcionar as alterações no ambiente em que se encontra e reagir, reflectindo essas alterações no próprio comportamento (processo referido frequentemente como aprendizagem).

Intencionalidade é a capacidade do agente assumir um objectivo de modo a que o seu próprio comportamento seja orientado no sentido de alcançar esse objectivo.

Convém realçar que a descrição de agente, e das suas características e propriedades, apresentada é geral. Em função do fenómeno que se pretende estudar e com base nas hipóteses teóricas consideradas, os agentes de um determinado modelo poderão apresentar algumas das características referidas mais desenvolvidas e outras menos: por exemplo, a capacidade de adaptação e de aprendizagem poderão não ser relevantes ou o agente poderá não ser orientado por um objectivo determinado. No entanto, algumas das propriedades e características referidas estarão sempre presentes.

Por outro lado, o agente actua num ambiente onde percepciona, interpreta e provoca as mudanças desse ambiente de acordo com as regras definidas e implementadas para o efeito. Em geral, este ambiente pode ser:

- Acessível (em maior ou menor nível);
- Determinístico;
- Estático;
- Contínuo.

O ambiente é tanto mais acessível quanto mais informação sobre este o agente conseguir obter. E quanto mais detalhada for esta informação, mais eficaz será o agente nas acções que toma no sentido de atingir o objectivo desejado. No entanto, só em casos ideias, o ambiente será completamente acessível, ou seja, o agente obtém toda a informação, com todo o detalhe necessário, sobre o estado do ambiente em que se encontra.

Um ambiente determinístico é aquele em que uma dada acção por parte do agente resulta sempre na mesma mudança de estado do ambiente, ou seja, não haverá incerteza sobre o resultado da acção tomada pelo agente. Esta situação verifica-se só em casos ideais por dois motivos principais:

- Existem outros eventos que afectam o estado do ambiente para além das acções do agente;
- As acções que o agente toma podem não ter o efeito previsto e desejado.

Um ambiente estático muda somente em consequência das acções tomadas pelo agente. Outra característica que na realidade dificilmente existe, pois existem inúmeros eventos externos que provocam alterações no ambiente, eventos esses que, por sua vez, são muitas vezes provocados por outros agentes que actuam no mesmo ambiente.

Um ambiente contínuo, em contraposição a um ambiente discreto, é aquele em que existem um número infinito de estados possíveis. Quanto menor for o número de estados possíveis do ambiente, maior será a eficácia do agente ou tanto mais será este simples assumindo-se uma eficácia constante.

Assim, em geral e a um nível abstracto, a definição de um Modelo Baseado em Agentes contempla os seguintes aspectos principais (Gilbert et al., 2000):

1. Definir as propriedades cognitivas e sensoriais dos vários tipos de agentes envolvidos no modelo face aos outros agentes e ao ambiente em que se encontram;

2. Definir as acções que o agente poderá tomar e os critérios que serão utilizados pelo agente para escolher a acção a tomar;

3. Definir as propriedades do ambiente em que os agentes se encontram.

Após a criação do modelo que implemente estas definições, este é executado (ou seja é posto a "correr") de modo a que seja possível observar os resultados. Através destes resultados, o investigador extrai conclusões sobre o fenómeno estudado com base na identificação de estruturas e padrões emergentes, ou seja, observados ao nível do sistema como um todo, estruturas e padrões esses que não são em geral possível identificar a partir do conhecimento do comportamento de cada tipo de agente.

3.2. Principais Características

De acordo com Nigel Gilbert, as características relevantes dos Modelos Baseados em Agentes são (Gilbert, 2007):

- Correspondência ontológica;
- Heterogeneidade dos agentes;
- Representação do ambiente;
- Interacção entre os agentes;
- Racionalidade limitada;
- Aprendizagem;

A associação entre os agentes do modelo e as entidades envolvidas no fenómeno a estudar, denominada "Correspondência Ontológica", facilita tanto a criação do modelo como também a interpretação dos resultados.

O Modelo Baseado em Agentes permite representar de forma distinta as várias entidades intervenientes no fenómeno, com maior ou menor granularidade, considerando diferentes regras de comportamento, o que nos modelos analíticos é muitas vezes impossível de conseguir.

Uma representação eficaz do ambiente em que os agentes actuam, contemplando adequadamente os aspectos relevantes do fenómeno estudado, é outra das vantagens mais significativas dos Modelos Baseados em Agentes.

Outra vantagem significativa dos Modelos Baseados em Agentes é a possibilidade de simular os vários possíveis modos de interacção entre os agentes, permitindo a implementação de mecanismos de comunicação que reproduzam bastante fielmente os mecanismos existentes no fenómeno estudado. Em relação a este aspecto, há que considerar que a criação das relações entre os agentes participantes nos fenómenos sociais é de natureza endógena, ou seja são os próprios agentes que decidem, directa ou indirectamente, com quais outros agentes se relacionam, no sentido que cada agente decide autonomamente, com base na vantagem que prevê obter, quando iniciar ou terminar uma relação com outro agente. São mais raros, os modelos em que as relações devem ser estabelecidas exogenamente, ou seja, por alguma entidade externa ou como consequência de alguma característica do modelo como, por exemplo, a topologia do ambiente em que os agentes actuam (Vriend 2006).

Ao contrário do que tem sido feito em muitos modelos analíticos, sobretudo na área das Ciências Económicas, em que se considera que os agentes actuam de forma totalmente racional e com capacidade muito sofisticadas, certamente acima do que na realidade acontece,

é muito fácil implementar, nos Modelos Baseados em Agentes, o que se pode referir como a característica de "Racionalidade Limitada": ou seja, os agentes são criados com uma "*inteligência*" com capacidade semelhante à inteligência das entidades intervenientes do fenómeno real.

Por fim, nos Modelos Baseados em Agentes é possível simular a propriedade de "Aprendizagem", tanto ao nível do agentes individual, em que este aprende com a experiência adquirida, como ao nível evolutivo, em que surgem novas gerações de agentes mais capazes das anteriores, e bem como ao nível social, em que os processos de imitação provocam uma uniformização dos comportamentos individuais.

3.3. Utilização de Modelos Baseados em Agentes

Nas Ciências Económicas, os Modelos Baseados em Agentes mais comuns são modelos de mercados, de inovação, de conflitos ou dilemas sociais, de dinâmica da informação e de crescimento económico e convergência (Araújo, 2010).

Por outro lado, a utilização de Modelos Baseados em Agentes nas Ciências Sociais tem-se vindo a generalizar essencialmente devido à complexidade dos fenómenos que se pretendem estudar (Macal et al., 2005).

Os principais factores que provocaram, e continuam a provocar, a generalização na utilização dos Modelos Baseados em Agentes são os seguintes:

- A não adequabilidade de outros tipos de modelos no âmbito das Ciências Sociais devido às restrições impostas por esses modelos nos comportamentos dos agentes sociais e nas características dos ambientes em que os agentes actuam;

- Os fenómenos sociais a estudar são cada vez mais complexos, com inúmeras intra-dependências, o que torna muito difícil, e frequentemente mesmo impossível, a aplicação de métodos analíticos;
- Nos Modelos Baseados em Agentes, os agentes apresentam frequentemente muitos aspectos comuns com as entidades que representam, facilitando assim a aplicação dos conceitos e abstracções teóricas;
- No estudo dos fenómenos sociais, a emergência de padrões a partir da interacção entre os indivíduos que constituem os grupos sociais é frequentemente um aspecto fundamental e esta característica é especialmente bem estudada através de Modelos Baseados em Agentes;
- A capacidade, alcance e eficácia dos meios computacionais são cada vez maiores.

A utilização de Modelos Baseados em Agentes é recomendada quando se verificam algumas das seguintes situações (Macal et al., 2005):

- As entidades intervenientes no fenómeno a estudar são natural e facilmente representadas por agentes;
- As relações entre essas entidades são dinâmicas e o seu conhecimento é determinante para o estudo do fenómeno em questão;
- Existe a necessidade de considerar as características de adaptação e de aprendizagem das entidades intervenientes;
- É relevante a constituição de grupos das entidades envolvidas, formando organizações;
- A componente espacial tem impacto no comportamento e relacionamento das entidades intervenientes;

- Não é possível prever a evolução do sistema simplesmente com base na análise do comportamento do sistema no passado.

3.4. Fases do processo de investigação baseado na simulação

O processo de investigação científica fundamentada na técnica da simulação em geral, e nos Modelos Baseados em Agentes em particular, deve ter em conta quatro fases fundamentais (Axelrod 2005):

- A construção/programação do modelo;
- A análise dos dados obtidos a partir da execução do modelo;
- A apresentação dos resultados dessa análise;
- A validação do modelo.

Ainda de acordo com Axelrod, cada uma destas quatro fases estabelece requisitos a serem contemplados pelo programa que implementa o modelo.

3.4.1. 1ª Fase - Construção/Programação do modelo

A primeira fase (construção/programação do modelo) tem essencialmente três requisitos:

- A construção do modelo deve ser correcta no sentido do programa implementar correctamente o modelo independentemente de este representar mais correctamente ou menos correctamente a realidade;

- O programa que implementa o modelo deve estar ao alcance do investigador de modo a permitir que este entenda como o modelo foi implementado, como são definidos os valores de entrada e como devem ser interpretados os valores de saída;
- O programa deve poder ser alterado com facilidade de modo a implementar as alterações ao modelo consideradas oportunas, ou seja, o programa deve poder ser facilmente extensível.

No caso de Modelos Baseados em Agentes, a definição do modelo é feita através da perspectiva do agente, ao contrário do que acontece na criação de outros tipos de simulações em que a perspectiva é a do processo (Macal et al., 2005). Assim, muito em geral, as principais actividades realizadas na criação de um Modelo Baseado em Agentes são:

- Identificar os agentes e estabelecer hipóteses teóricas sobre o seu comportamento contemplando propriedades relevantes e regras de decisão;
- Identificar o ambiente em que os agentes actuam;
- Estabelecer hipóteses teóricas sobre como os agentes se relacionam entre si e com o ambiente;
- Implementar o modelo na plataforma escolhida;

3.4.2. 2ª Fase – Análise de Dados

Em relação à segunda fase, análise de dados, há que considerar, por um lado, o grande número de dados que se obtém na execução do modelo, e por outro lado, o facto que geralmente tem relevância a análise de dados históricos obtidos em execuções anteriores do

modelo com diferentes valores de entrada ou diferentes valores de semente das sequências aleatórias utilizadas.

3.4.3. 3ª Fase - Apresentação dos Resultados

A terceira fase, relacionada com a apresentação dos resultados, deve ter em consideração que, quase sempre, é difícil apresentar de forma sucinta os resultados de uma simulação completa devido aos seguintes factores:

- Os resultados dependem geralmente dos detalhes do modelo o que implica que é necessário descrever esses detalhes para que o modelo possa ser replicado;
- Geralmente, a descrição dos dados históricos exige uma exposição extensa por não haver ainda uma suficiente padronização dos conceitos.
- Dada a variedade dos destinatários da informação, geralmente é necessário explicar conceitos específicos a cada uma das áreas de conhecimento envolvidas.
- Por fim, considerando também que ainda não existe uma grande e difundida familiaridade com o método da simulação, será necessário descrever com detalhe como o modelo foi implementado, focando o alcance e as limitações do modelo.

3.4.4. 4ª fase - Validação do Modelo

Por fim, na quarta e última fase, o modelo deve ser validado no sentido de se confirmar que os resultados do modelo reproduzem os resultados que ocorrerão no cenário real. Existem essencialmente três métodos que podem ser utilizados para efectuar a validação do modelo e que devem na medida do possível ser aplicados complementarmente:

- Análise detalhada dos algoritmos implementados no modelo e das variáveis de entrada e de saída;
- Comparação dos resultados obtidos com os resultados observados no cenário real;
- Comparação dos resultados obtidos com os resultados fornecidos por outros modelos que reproduzam o mesmo fenómeno.

Há que ter presente as seguintes duas questões relacionadas com a validação do modelo (Gilbert 2004):

- Os fenómenos sociais apresentam geralmente características aleatórias reproduzidas através da obtenção de valores estocásticos nos modelos criados – portanto a coincidência dos resultados obtidos do mesmo fenómeno a partir de vários modelos terá que ser verificada aplicando métodos estatísticos;
- A coincidência dos resultados obtidos com um dado modelo com os dados reais (numa perspectiva estatística) não é uma condição suficiente, apesar de ser necessária, para a validação do modelo porque esses mesmos resultados poderão também ser obtidos utilizando modelos incorrectos em determinadas situações.

3.5. Abordagens e Arquitecturas

Após termos visto no ponto anterior, de uma forma genérica, quais são as principais actividades na construção de um Modelo Baseado em Agentes, será útil vermos agora os aspectos mais relevantes nas abordagens seguidas neste processo porque esses aspectos poderão condicionar a arquitectura que será adoptada para a plataforma aplicacional a desenvolver.

3.5.1. Abordagens

Não existe uma abordagem para a construção de Modelos Baseados em Agentes que seja reconhecida como a melhor, existindo abordagens diferentes em função do objectivo da simulação pretendida. No entanto, há que contemplar os aspectos que caracterizam os Modelos Baseados em Agentes (Gilbert et al., 2000), tendo em consideração que nem sempre todas essas características devam existir num dado modelo. Nomeadamente:

- O agente deve ser capaz de percepcionar o ambiente e as acções dos outros agentes;
- O agente deve poder ter memória, ou seja, o registo da informação recebida e das suas acções anteriores;
- O agente deve ser capaz de identificar a acção a tomar com base em regras definidas;
- O agente deve poder actuar sobre o ambiente e os outros agentes.

Em geral, mas agora com maior detalhe do que foi exposto no ponto anterior, o desenvolvimento de um modelo é constituído pelas seguintes actividades (Gilbert 2004):

- Definição do objectivo da simulação e da teoria a aplicar;
- Desenho das entidades intervenientes no modelo;
- Desenho do ambiente em que as entidades actuam;
- Definição do comportamento dinâmico do modelo;
- Definição da interface de utilizador;
- Implementação do modelo;

- Verificação da implementação.

A definição do objectivo da simulação deve ser fundamentada na questão de investigação: esta deve ser claramente formulada, eventualmente decompondo-a em outras questões de investigação parciais com um âmbito mais restrito, para que seja possível identificar claramente as entidades que terão que previstas no modelo a construir.

Por outro lado, a identificação das possíveis teorias aplicáveis ao estudo do fenómeno em questão, considerando os aspectos relacionados com a dinâmica do fenómeno e os processos de mudança envolvidos, dá pistas muito importantes acerca dos conceitos a ter em conta (e, consequentemente, acerca das entidades que deverão ser modelizadas), e acerca das hipóteses que deverão ser contempladas no modelo a implementar.

Após o objectivo do modelo estar bem definido, contemplando os conceitos e as entidades a modelizar, e após as hipóteses a considerar ficarem bem identificadas, é possível passar à actividade seguinte: o desenho do modelo. Nesta actividade, são definidos os vários tipos de objectos que deverão ser implementados, especificando os respectivos atributos, de modo a modelizar todas as entidades identificadas. A definição destes vários tipos de objectos deverá resultar numa estrutura hierárquica de classes com as classes mais genéricas no topo e as restantes derivando destas de acordo com os princípios fundamentais da programação orientada por objectos.

A próxima actividade é o desenho do ambiente em que as entidades intervenientes no modelo actuam: este ambiente poderá ser um espaço dimensional (com uma, duas ou três dimensões) tendo cada entidade, neste caso, os atributos necessários para definir a própria posição, ou poderá ser uma rede de conexões estabelecidas de acordo com as formas de relacionamentos relevantes no fenómeno estudado.

Após a definição das características estáticas do modelo há que identificar as características dinâmicas, através da definição dos algoritmos que representam a interacção dos vários agentes entre si e com o ambiente. A implementação destes algoritmos será efectuada ao nível dos métodos dos vários tipos de objectos identificados.

O desenho da interface de utilizador deverá considerar todas as variáveis de entrada e de saída do modelo, identificando o tipo de objecto gráfico mais apropriado para representar cada uma destas variáveis. Há que considerar nesta actividade também a necessidade de serem utilizados outros meios para a passagem dos dados para o modelo e para a extracção de dados dos modelos, como por exemplo, ficheiros de dados (hoje em dia, geralmente em formato XML).

A este ponto o modelo está definido e pode ser implementado utilizando o ambiente de programação escolhido. A implementação do modelo consiste essencialmente na personalização dos componentes da plataforma utilizada e na codificação do que for necessário programar especificamente. A personalização terá mais ou menos peso da programação em função das características da plataforma escolhida.

Geralmente, em paralelo ao trabalho de implementação do modelo deve decorrer a implementação dos programas de teste necessários para verificar a correcta implementação. O ideal será obter um conjunto de programas de teste completos que permitam verificar, de forma automática, qualquer alteração efectuada na implementação do modelo. Estes programas de teste deverão contemplar vários valores de entrada e os correspondentes valores de saída que se sabem que são correctos de modo a verificar a correcção do modelo implementado.

3.5.2. Arquitecturas

Uma das arquitecturas usadas para criar Modelos Baseados em Agentes consiste num motor interpretador de regras, onde cada regra é constituída essencialmente por duas partes: uma condição e uma acção que é executada quando a condição for verdadeira. Em cada ciclo, o motor executa as regras definidas e modifica (e guarda) o estado do modelo constituído pelo conjunto de propriedades que caracterizam os vários agentes e o ambiente. Neste caso, o modelo é construído definindo o conjunto de regras a executar: condições e acções correspondentes.

Este tipo de arquitectura permite construir com bastante facilidade Modelos Baseados em Agentes em que os agentes respondem aos estímulos recebidos de acordo com um algoritmo pré-definido.

Para os casos em que se pretende que o agente altere o próprio algoritmo de reacção aos estímulos com base nos resultados obtidos (algoritmos adaptativo), este tipo de arquitectura é menos adequado. Para este tipo de simulação, em que se pretende que o agente altere a forma de reagir aos estímulos com base na história das interacções, ou seja, em que se pretende que o agente aprenda a se adaptar à mudança do que o envolve, existem duas técnicas fundamentais:

- Modelos baseados em Redes Neuronais;
- Algoritmos Genéticos.

Ao contrário do que se passa nos sistemas baseados em motores de regras, as redes neuronais modificam as respostas aos estímulos externos com base na experiência realizada: esta adaptação surge porque os coeficientes utilizados nas expressões que trabalham sobre os estímulos recebidos são modificados, através da aplicação de casos de referência, de modo a

que os resultados finais estejam o mais possivelmente conformes com o esperado. Várias topologias de rede têm sido experimentadas com o objectivo de encontrar quais as mais adequadas para a criação de capacidades de aprendizagem eficazes.

A outra arquitectura que permite a criação de agentes com capacidade de aprendizagem é baseada nos chamados Algoritmos Genéticos que seguem uma orientação biológica. Neste tipo de modelo, existe uma métrica do nível de adequação do agente ("fitness degree") ao ambiente em que se encontra de forma que os agentes que possuem uma melhor adequação também possuirão maior capacidade reprodutiva, dando origem a uma nova geração de agentes que terão, assim, uma maior probabilidade de melhor se adaptarem ao ambiente. Por outro lado, no processo de reprodução são introduzidas alterações aleatórias, inspiradas na ideia de mutação, o que proporciona variações nas novas gerações de forma a contribuir para obter uma melhor adaptabilidade das mesmas. Este processo repetido ao longo de vários ciclos geracionais, dá origem a uma população cada vez mais capaz de obter melhores resultados no ambiente em que se encontra.

4. Quadro de Referência Conceptual

O quadro de referência conceptual a considerar na construção de Modelos Baseados em Agentes será indispensável para a definição da arquitectura da plataforma a desenvolver de modo a que esta plataforma contemple adequadamente os vários tipos de modelos que poderão ser construídos sobre ela.

Este quadro de referência deverá ser criado tendo em conta todas as entidades envolvidas nos Modelos Baseados em Agentes, as suas propriedades e como estas entidades se relacionam entre si. Assim, o quadro de referência conceptual que servirá de orientação ao desenho da plataforma aplicacional contempla dois aspectos fundamentais resultantes das características dos Modelos Baseados em Agentes e da sua utilização (Axtell, 2000):

- Entidades envolvidas;
- Processo de execução.

Considerando que a plataforma aplicacional a desenvolver será construída utilizando uma linguagem orientada por objectos, de acordo com o descrito no capítulo seguinte, utilizar-se-ão termos próprios deste paradigma para descrever aqueles aspectos. Assim, obter-se-á uma maior clareza e precisão na exposição dos conceitos apresentados.

4.1. Entidades Envolvidas

O elemento fundamental da estrutura geral dos Modelos Baseados em Agentes é o objecto agente ("*Agent*"), o qual possui propriedades que descrevem o seu estado e métodos que descrevem as regras que definem o seu comportamento. Tanto o estado como o

comportamento do agente podem ser privados ou públicos: no primeiro caso só são visíveis ao próprio agente; no segundo caso são visíveis aos outros agentes e, em geral, aos outros objectos constituintes o modelo.

Em vários tipos de Modelos Baseados em Agentes, o estado do agente poderá ser representado por uma única propriedade, por sua vez também um objecto, que é percepcionada pelos outros agentes e que determina a interacção entre os agentes, podendo-se assumir que esta propriedade representa a morfologia do agente ("*Morphology*").

A população ("*Population*") constituída por todos os agentes do modelo, ou por um grupo de agentes afins (neste caso o modelo será constituído por várias populações), é também um objecto com as suas propriedades e métodos próprios (privados ou públicos). Neste caso, os métodos poderão ser utilizados para obter, por exemplo, dados estatísticos característicos da população.

Outra entidade fundamental é o ambiente ("*Environment*") em que actuam os agentes. O ambiente é um objecto com propriedades específicas, as quais, por um lado, são percepcionadas pelos agentes, e por outro lado, podem ser alteradas como resultado da actuação desses agentes. Uma das propriedades do ambiente que desempenha um papel fundamental em alguns tipos de Modelos Baseados em Agentes é a matriz de posicionamento ("*Grid*"). Neste caso, cada agente terá uma posição vem definida, representada por um elemento da matriz ("*Cell*"), e poderá deslocar-se de acordo com a topologia da referida matriz.

Por fim, poder-se-á considerar uma entidade global contendo todas estas entidades, representando o modelo na sua globalidade ("*World*").

O seguinte diagrama de Entidades – Associação representa as várias entidades acima referidas e como estas se relacionam entre si.

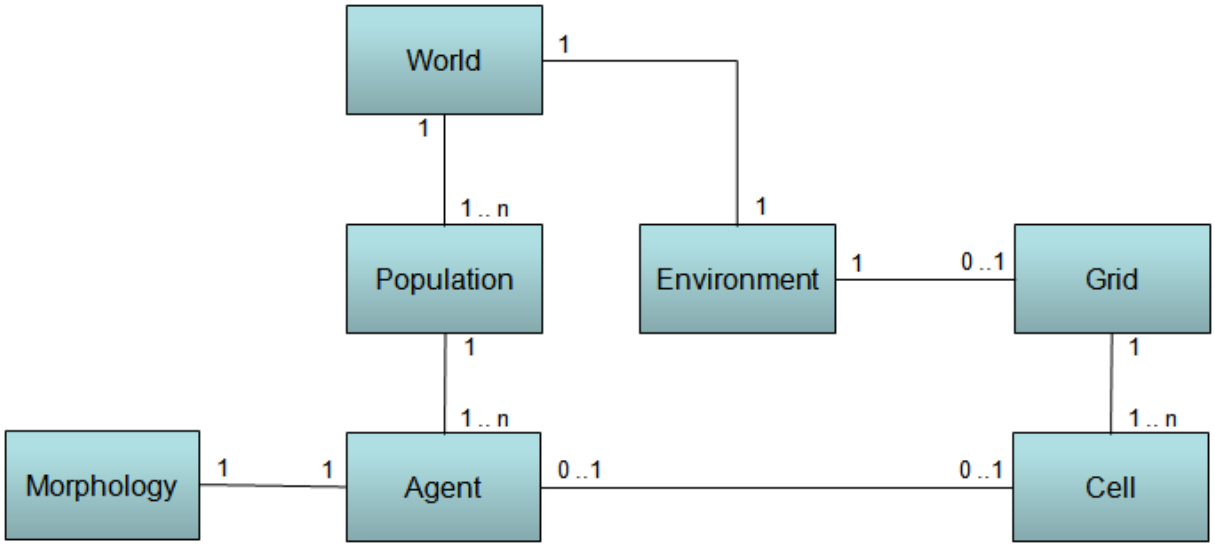

Figura 1 – Diagrama Entidades - Associação

4.2. Dinâmica dos Modelos Baseados em Agentes

Considerando a descrição, apresentada na primeira parte deste capítulo, de como os Modelos Baseados em Agentes funcionam, pode-se esquematizar o processo de execução de um modelo genérico como constituído por seis passos, conforme apresentado no seguinte esquema (Araújo, 2010):

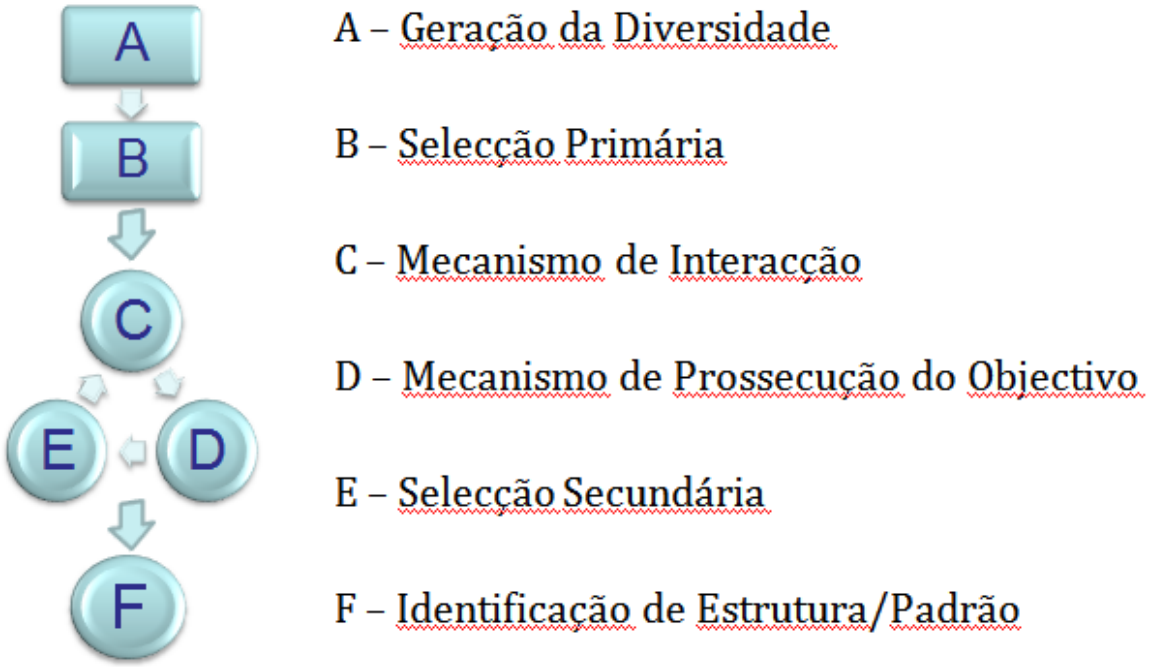

Figura 2 – Dinâmica dos Modelos Baseados em Agentes

O ciclo de execução propriamente dito é constituído por três passos ("Mecanismo de Interacção", "Mecanismo de Prossecução do Objectivo" e "Selecção Secundária"). Antes deste ciclo são executados dois passos de preparação ao ciclo de execução ("Geração da Diversidade" e "Selecção Primária") e na saída do ciclo é executado o passo "Identificação de Estrutura/Padrão".

Geração da Diversidade

No primeiro passo é criada, geralmente de forma aleatória, uma ou mais populações de agentes cujas propriedades assumem valores variados.

Selecção Primária

Após a criação da diversidade inicial, é efectuada uma primeira selecção de agentes baseada nas propriedades destes, ou de acordo com a estrutura de rede em que estes se encontram, ou aleatoriamente.

Mecanismo de Interacção

Neste passo os agentes interagem entre si (Sociabilidade) e, de acordo com o modelo específico, podem também interagir com o ambiente (Adaptabilidade). A interacção dos agentes entre si poderá ser determinada por questões relacionadas com a afinidade ou semelhança dos agentes entre si, e poderá ser realizada entre todos os agentes, ou somente entre os agentes escolhidos aleatoriamente, ou entre agentes que satisfazem determinadas condições preestabelecidas contemplando uma ou mais propriedades dos agentes.

Mecanismo de Prossecução do Objectivo

No segundo passo do ciclo de execução, contempla-se o facto que, num Modelo Baseado em Agentes, os agentes podem orientar o seu comportamento (Autonomia) no sentido de alcançar um determinado objectivo (Intencionalidade). Frequentemente, o agente escolhe o seu comportamento, ou seja toma uma decisão, aplicando uma "Função de Utilidade" que permita medir o resultado da decisão em termos de prossecução do objectivo, ou identificando a melhor estratégia para lidar com o ambiente em que se encontra.

Selecção Secundária

No último passo do ciclo de execução são seleccionados os agentes que permanecerão no modelo, ou seja que farão parte da próxima geração, com base num critério preestabelecido

(relacionado com a "Função de Utilidade" ou com o nível de adaptação do agente ao ambiente).

Identificação de Estrutura/Padrão

A conclusão do ciclo de execução é determinada de acordo com um critério preestabelecido. Este critério pode ser muito simplesmente atingir um número preestabelecido de ciclos; no entanto, mais frequentemente, o ciclo de execução é concluído quando o modelo assume uma configuração permanente ou o modelo passa repetidamente pelas mesmas configurações (neste contexto, a configuração do modelo é determinada pelo estados de todos os agentes que o constituem). Nesta situação, a grande diversidade inicial é significativamente reduzida e será possível identificar estruturas ou padrões emergentes.

5. O Contexto Tecnológico

A construção da plataforma aplicacional foi efectuada no contexto do ambiente Windows .Net da Microsoft. Neste capítulo são apresentados os aspectos mais relevantes relacionados com esta escolha.

5.1. Plataformas específicas para desenvolvimento de Modelos Baseados em Agentes

Existem inúmeras plataformas com bibliotecas de programas e ferramentas específicas para o desenvolvimento de Modelos Baseados em Agentes (Railsback et al. 2006), sendo muitas delas constituídas por um quadro de referência conceptual ("*framework*") descrevendo e enquadrando os conceitos principais a utilizar e por bibliotecas de programas que implementam de forma genérica estes conceitos.

Entre estas inúmeras plataformas, convém referir, por serem as mais divulgadas, as seguintes:

- Swarm
- Repast
- Mason
- NetLogo

A plataforma Swarm (Minar et al., 1996) é um laboratório computacional genérico, para vários tipos de modelos em várias áreas de pesquisa, constituído por um conjunto de ferramentas específicas criadas com as linguagens de programação Object C e Java e que disponibiliza também meios para a criação de laboratórios virtuais para observação e

experimentação sobre os modelos criados. Uma das principais características da plataforma Swarm é a possibilidade de definir agentes de forma iterativa, ou seja agentes cujo comportamento é o resultado de um modelo de agentes de nível mais elementar, e assim, sucessivamente.

A plataforma Repast, com origem na plataforma Swarm, é um conjunto de ferramentas criadas para ambientes Java, Windows .Net e Python Scripting para a construção de modelos na área das Ciências Sociais.

A plataforma Mason (Luke et al., 2004) é uma livraria Java constituída essencialmente por uma livraria genérica, compacta e com muito bom desempenho, com o foco na criação de modelos constituídos por um número muito grande de agentes e contemplando a necessidade de execução em paralelo de um grande número de simulações em várias plataformas de HW e SW.

Estas três plataformas apresentam arquitecturas e utilizam linguagens que se podem considerar *standards*, podendo ser utilizadas para a implementação de modelos complexos. No entanto, apresentam dificuldades na sua utilização, sobretudo ao nível da visualização e ao nível da compilação dos resultados.

Para além destes três ambientes, há que referir também uma quarta plataforma, NetLogo, que foi criada para ensinar os conceitos básicos da modelização através de uma linguagem específica de muito fácil utilização por quem não tenha conhecimentos de programação e que tem vindo a ser utilizada, cada vez mais, na criação de modelos com alguma complexidade. No entanto, por estar direccionado para a criação de modelos com os agentes situados em matrizes de células e por utilizar uma linguagem de programação específica, o NetLogo tem sido considerado limitado pela comunidade científica, apesar de ser uma plataforma muito

fácil de utilizar, para além da apresentação e da documentação que são de muito boa qualidade.

O nível de versatilidade de cada plataforma, medido pela capacidade de implementar vários tipos de modelos, é geralmente reduzido com o aumentar da facilidade com que os modelos são construídos. Ou seja quanto mais o autor utilizar um ambiente de desenvolvimento mais específico para o tipo de modelo a que se destina, menor será a flexibilidade e a liberdade na implementação de requisitos não explicitamente previstos na plataforma.

5.2. Ambientes e Linguagens de Programação Genéricos

Apesar das plataformas específicas apresentarem certamente uma maior facilidade na criação de novos modelos, muitas vezes também limitam também a possibilidade de implementar novas abordagens.

Existe uma alternativa a este tipo de plataformas que também deve ser considerada: a utilização de ambientes e linguagens de programação genéricos. Neste caso, o modelo é criado praticamente a partir do zero, no sentido que o ambiente não tem componentes específicos para serem utilizados em Modelos Baseados em Agentes.

Esta opção poderá ser considerada como o extremo oposto da opção de utilização de uma plataforma específica para Modelos Baseados em Agentes: máxima flexibilidade mas com uma muito maior dificuldade de implementação derivante do facto que não existe nem uma estrutura predefinida a usar nem são disponibilizados componentes funcionais específicos.

Exemplos destes ambientes de programação são as linguagens MATLAB e Mathematica (vocacionadas para a computação numérica e para desenvolvimento de programas na áreas da Engenharia e das Ciências Matemáticas e Físicas), bem como linguagens de programação

totalmente genéricas frequentemente utilizadas para a criação de programas para o mundo empresarial (linguagens C, C++, C#, Visual Basic, Java, e outras).

5.3. Plataformas Aplicacionais Genéricas

Existe uma via intermédia entre os dois extremos opostos referidos nos dois pontos anteriores (Plataformas Específicas e Linguagens de Programação Genéricas) a qual consiste na utilização do que se poderá denominar como Plataforma Aplicacional Genérica. Trata-se de uma "*framework*" aplicacional em que existe uma estrutura e conceitos de base predefinidos a utilizar no desenvolvimento de novos Modelos Baseados em Agentes mas que permite com a máxima flexibilidade construir os aspectos específicos do novo modelo. Assim, este último tipo de plataforma pretende disponibilizar o melhor compromisso entre as duas exigências de fundo: facilidade de implementação e flexibilidade.

Outra característica deste tipo de plataforma é que só exige conhecimento de uma linguagem de programação comum e padronizada, não exigindo a aprendizagem de uma ferramenta específica para criação de Modelos Baseados em Agentes. Este facto poderá ser uma vantagem ou desvantagem, consoante as competências de quem tiver que criar o modelo.

Um bom exemplo desta abordagem é a plataforma SimBioSys (McFadzean, 1994) especificamente criada para o desenvolvimento de modelos com algoritmos de natureza biológica/genética.

Assim, com base nas considerações anteriores foi tomada a opção de criar, no âmbito do presente trabalho, uma plataforma aplicacional genérica, utilizando, como já referido, o ambiente Windows .Net da Microsoft e a linguagem de programação C#.

5.4. O ambiente de programação

Tendo em vista o entendimento, a verificação e a alteração, por parte de outros investigadores, dos modelos produzidos, o ambiente de programação, nomeadamente as ferramentas utilizadas e os padrões de codificação e documentação aplicados, é de fundamental importância na qualidade dos modelos produzidos (Gilbert et al., 2000).

A melhor forma de garantir este objectivo é a utilização de uma linguagem orientada por objectos que permite traduzir o fenómeno que se pretende estudar, representando os agentes por objectos com propriedades específicas que reagem a eventos externos através de métodos predefinidos. Para além de representar os agentes por objectos, também se representarão outras entidades intervenientes no modelo por objectos específicos (por exemplo, a população de agentes e o ambiente). Ou seja, tudo o que tem relevância para o modelo é representado através do conceito de objecto, com propriedades e métodos específicos do comportamento a ser modelizado.

5.5. Linguagens orientadas por objectos

O conceito fundamental numa linguagem orientada por objectos é naturalmente o ‘‘objecto’’, o qual deve ser entendido como uma unidade conceptual constituída por um conjunto de propriedades ou atributos que representam o estado do objecto e por um conjunto de métodos que representam o comportamento dinâmico do objecto.

O comportamento do objecto é implementado pela classe que representa o tipo de objecto e o estado do objecto é contido em cada objecto que assim deve ser considerado como uma instância da respectiva classe. Por exemplo, o conceito de agente corresponde a um tipo específico de objecto que é definido por uma classe específica onde se encontram definidas as

propriedades que caracterizam o estado do agente (por exemplo a sua morfologia, por sua vez também um objecto definido por uma classe específica), bem como os métodos que traduzem os possíveis comportamentos do agente (por exemplo, a forma como o agente interage com outros agentes).

Uma linguagem orientada por objectos implementa pelo menos três características fundamentais do paradigma em questão: o encapsulamento, a herança e o polimorfismo.

O conceito de encapsulamento representa a possibilidade de não deixar que determinadas propriedades ou determinados métodos de uma classe possam ser visíveis a partir de outras classes da mesma aplicação. O encapsulamento permite fazer evoluir uma classe sem ser necessário alterar o resto da aplicação em que esta classe é utilizada.

O conceito de herança tem por base os conceitos de classe mãe e classe filha, de forma a permitir derivar uma nova classe a partir de outra classe existente modificando na classe filha, tanto as propriedades que representam o estado, como os métodos que representam o comportamento, evitando tanto quanto possível a duplicação de código.

O polimorfismo está associado ao conceito de "herança" porque representa a utilização de uma classe-filha em código que utiliza a classe-mãe, o que permite uma codificação mais genérica dos algoritmos implementados na aplicação

Polimorfismo representa a possibilidade de utilizar um objecto, instância de uma determinada classe com uma dada classe-mãe, em código onde são utilizados objectos que são instâncias da classe-mãe. Esta possibilidade permite uma codificação mais genérica, ou seja, permite uma maior flexibilidade na aplicação. Por exemplo, havendo a necessidade de definir uma classe que represente um agente com alguma propriedade específica, poder-se-á definir esta classe como classe-filha da classe mais genérica que representa agentes em geral. Assim, objectos que representem o agente em questão poderão ser utilizados no código em que sejam

utilizados agentes do tipo da classe-mãe (por exemplo, uma rotina que implemente um processo de interacção entre agentes).

5.6. O ambiente .Net da Microsoft

Os dois ambientes mais comuns para a construção de aplicações de natureza empresarial em computadores pessoais são o ambiente .Net da Microsoft e o ambiente Java criado pela Sun Microsystems.

Estes ambientes, que têm muitos aspectos semelhantes entre si, disponibilizam funcionalidades básicas e genéricas que podem ser estendidas de modo a cumprir os requisitos específicos da aplicação a desenvolver, e disponibilizam várias ferramentas de programação sofisticadas que permitem o desenvolvimento eficiente de aplicações.

Não fazem parte do âmbito da presente dissertação, a comparação destes dois ambientes entre si e a análise das vantagens e desvantagens de cada um, tendo sido simplesmente escolhido o ambiente .Net da Microsoft para o desenvolvimento da plataforma aplicacional porque este é o ambiente com o qual o candidato se encontra mais familiarizado.

Convém, no entanto, realçar que os conceitos apresentados se mantêm inalterados nos dois ambientes e que a conversão da plataforma aplicacional desenvolvida para o ambiente Java é relativamente fácil.

6. A Plataforma Aplicacional

O presente capítulo tem o objectivo de descrever de forma bastante detalhada a arquitectura da plataforma aplicacional relacionando-a com a arquitectura de um modelo específico construído sobre a plataforma.

Assim pretende-se dar uma visão pormenorizada de como a plataforma está estruturada tendo em vista não só a sua utilização para a construção de Modelos Baseados em Agentes concretos mas também a extensão da plataforma de forma a contemplar requisitos funcionais não contemplados na versão inicial.

O capítulo é constituído essencialmente por três partes principais:

- Uma visão da arquitectura global;
- Uma descrição detalhada da arquitectura da plataforma aplicacional;
- Uma descrição da arquitectura de um modelo genérico focando exclusivamente aspectos relacionados com a plataforma aplicacional.

No fim do capítulo são apresentados alguns aspectos gerais que fazem parte da implementação da plataforma mas por não terem relevância sob o ponto de vista da arquitectura não foram referidos nos pontos acima mencionados.

6.1. Arquitectura Global

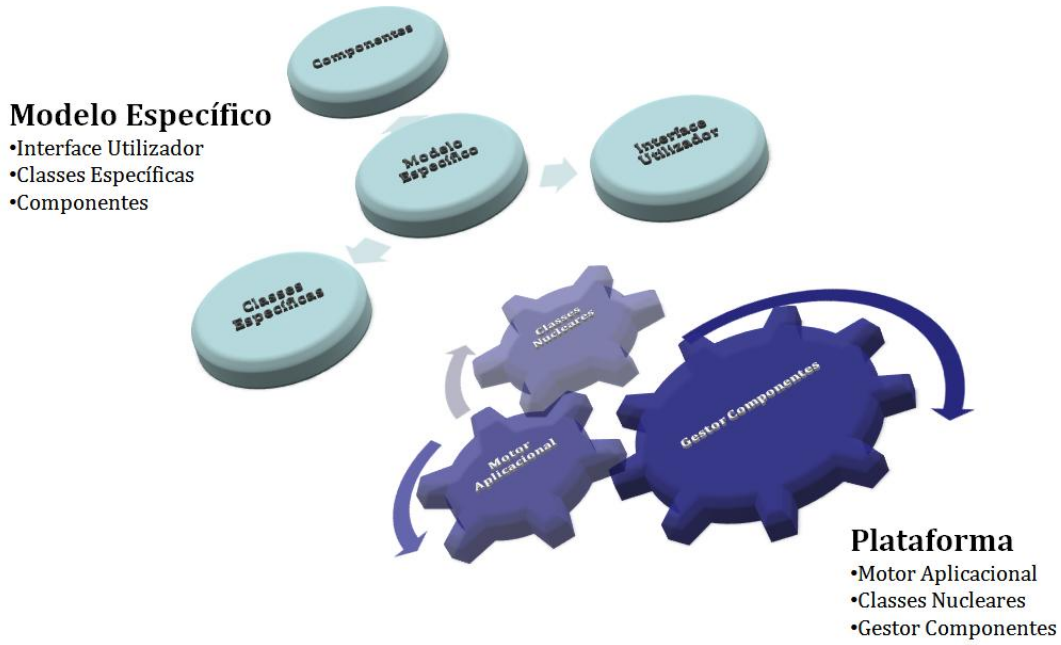

Figura 3 - Arquitectura Global da Plataforma Aplicacional

Nesta figura, mostra-se como a plataforma aplicacional construída se relaciona com um dado modelo específico e quais são os elementos principais que constituem tanto a arquitectura da plataforma aplicacional propriamente dita, como a arquitectura do modelo específico.

A ideia subjacente a esta arquitectura é que a plataforma aplicacional deve ser adequada para correr qualquer tipo de modelo e que, para isso, deve somente contemplar os conceitos gerais definidos no âmbito do quadro de referência conceptual descrito anteriormente.

Os aspectos específicos de um dado modelo são implementados fora da plataforma aplicacional, no âmbito do modelo específico, através da definição de componentes funcionais específicos que cumprindo uma estrutura geral são executados pelo motor

residente na plataforma e utilizando classes específicas derivadas das classes nucleares definidas na plataforma.

Esta arquitectura é modular e evolutiva: o âmbito de cada elemento da arquitectura está bem definido e limitado, sendo possível enriquecer progressivamente as funcionalidades contempladas em futuras versões da plataforma.

Existe também um outro benefício relevante resultante da arquitectura escolhida: dado que a estrutura dos vários modelos específicos é também modular, cumprindo requisitos comuns resultantes da necessidade da utilização da mesma plataforma aplicacional, é possível, e recomendável, reutilizar módulos já utilizados em modelos anteriormente desenvolvidos na criação de novos modelos. Esta vantagem foi muito significativa na construção dos modelos de exemplo descritos num capítulo mais adiante, sobretudo ao nível da interface de utilização.

6.2. Arquitectura da Plataforma Aplicacional

A arquitectura da plataforma aplicacional é constituída essencialmente por três entidades fundamentais:

- Motor Aplicacional;
- Classes Nucleares;
- Gestor de Componentes.

O Motor Aplicacional é uma classe (“Engine”), que possui um único método (“Run”) que implementa o ciclo de execução do modelo.

As Classes Nucleares implementam os conceitos fundamentais do quadro de referência adoptado para descrever a arquitectura geral de um Modelo Baseado em Agentes, por exemplo: a entidade Agente, a entidade Ambiente, a entidade População, a entidade Mundo, etc.

O Gestor de Componentes ("Plugins Manager") tem duas responsabilidades principais: carregar os componentes do modelo específico e executar esses componentes sob a supervisão do motor da plataforma.

6.2.1. Motor Aplicacional

A classe que implementa o motor da plataforma é constituída por um único método denominado "Run" que, depois de chamar o componente que inicializa o modelo ("InitWorld"), corre indefinidamente o ciclo de execução do modelo até ser verificada alguma condição que indique o fim desse ciclo. Após a saída do ciclo, o motor executa o componente "FinalizeWorld" onde são realizadas as acções necessárias para a correcta finalização da simulação, por exemplo apresentação dos valores finais.

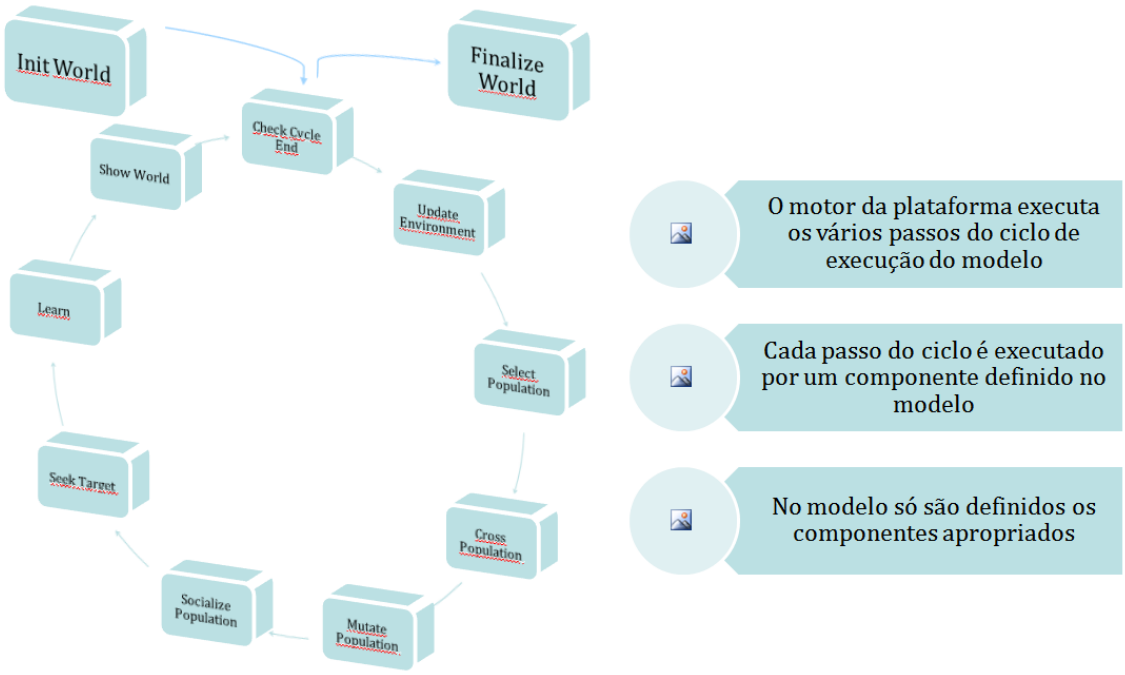

Figura 4 – O Motor Aplicacional

Em geral, os componentes que são executados repetidamente, no âmbito do ciclo de execução do modelo, são:

- "Update Environment"
- "SelectPopulation"
- "CrossPopulation"
- "MutatePopulation"
- "SocializePopulation"
- "SeekTarget"
- "Learn"

- "ShowWorld"

É importante realçar que os componentes que efectivamente irão estar presentes na implementação de um dado modelo, dependem do próprio modelo, ou seja, dependem da modelização que se considera apropriada para o fenómeno que se deseja estudar.

Em seguida, descrevem-se sumariamente cada um destes componentes, referindo o objectivo e âmbito do componente.

Componente “Update Environment”

Este componente actualiza o ambiente do modelo, efectuando as alterações que o modelo prevê realizar em cada ciclo; por exemplo, poderão ser alteradas algumas das propriedades globais do ambiente ou poderão ser efectuadas actualizações aos elementos pertencentes ao ambiente (por exemplo, obstáculos nos modelos espaciais).

Componentes "SelectPopulation", "CrossPopulation" e “MutatePopulation”

Nos modelos com características evolutivas (de inspiração biológica), em geral, existirão as três fases: a fase de selecção, a fase de reprodução e a fase de mutação.

Na fase de selecção, são seleccionados os agentes com melhor nível de adaptação, de acordo com a parametrização do modelo, sendo o nível de adaptação medido por uma função específica, geralmente denominada função de utilidade, com base nos valores de algumas propriedades de acordo com o estabelecido no modelo. Os agentes, cujo nível de adaptação ficar abaixo do limiar configurado, são eliminados.

Seguindo a analogia genética, a fase seguinte será constituída pelo cruzamento entre os agentes seleccionados, cruzamento este que dará origem a novos agentes que herdam algumas propriedades e comportamentos dos progenitores.

Por fim, na fase de mutação, são aleatoriamente introduzidas alterações nas propriedades e comportamentos de alguns agentes. Na vida real, a mutação faz parte intrínseca da fase de reprodução onde se realiza o cruzamento entre os progenitores; no entanto, é benéfico para a versatilidade do motor da plataforma considerar a mutação como uma fase separada a seguir à fase de cruzamento.

A cada uma destas três fases corresponde um componente específico.

Componente "SocializePopulation"

Neste componente, contemplam-se as eventuais acções, de acordo com o definido pelo modelo específico, de troca de informação entre agentes com o objectivo de traduzir fenómenos de interacção de natureza essencialmente social como, por exemplo, mecanismos de influência social.

Componente "SeekTarget"

Muitas vezes, a procura de um determinado objectivo por parte dos agentes é um factor fundamental do fenómeno social a estudar. Este aspecto, no âmbito da plataforma aqui descrita, deverá ser implementado no componente específico denominado "SeekTarget".

Componente "Learn"

A aprendizagem é outra característica importante em vários Modelos Baseados em Agentes: através da aprendizagem, o agente vai aperfeiçoando, ao longo do tempo, o seu

comportamento no sentido de obter melhores resultados. No componente "Learn" ficarão implementados os vários algoritmos de aprendizagem dos vários tipos de agentes.

Componente "ShowWorld"

Em cada ciclo de execução, geralmente haverá a necessidade de actualizar a apresentação do estado do modelo ao utilizador, recorrendo à interface de utilização prevista para o modelo ou à escrita em ficheiros de dados de saída conforme o desejado. É também neste componente que geralmente se adicionam os registos oportunos ao ficheiro de "Log" do modelo em que se regista toda a informação relevante descrevendo a história do modelo.

6.2.2. Gestor Componentes

A plataforma aplicacional tem uma classe específica que contempla todas as acções necessárias para a gestão dos componentes descritos no ponto anterior e que são utilizados para a implementação do comportamento do modelo ao nível macro. Ou seja, pode-se pensar que um dado modelo é construído implementando os algoritmos apropriados nos componentes que tenham um significado conceptual para o modelo em questão. Por exemplo, um modelo em que se pretenda definir agentes com um determinado objectivo, implementar-se-á no componente "SeekTarget" o algoritmo que, ao nível global do modelo, define como os agentes procedem para atingir o objectivo. Ou, outro exemplo, se num dado modelo, os agentes evoluem de acordo com um algoritmo genético, utilizar-se-ão os componentes "SelectPopulation", "CrossPropulation" e "MutatePopulation" para implementar as três fases geralmente existentes num algoritmo genético: selecção, cruzamento e mutação.

Esta implementação ao nível dos componentes é complementada pela criação das classes específicas ao modelo em questão em que se definem as propriedades e métodos específicos

ao nível de cada uma das entidades constituintes o modelo. Este aspecto é descrito mais adiante quando se apresentar o tópico "Classes Específicas" no âmbito da descrição da arquitectura de um modelo específico.

A plataforma terá que fazer o tratamento generalizado de um qualquer componente, contemplando essencialmente o carregamento e inicialização do componente e a execução do componente. Para que seja possível este tratamento generalizado, a estrutura de qualquer componente é única e bem definida conforme o descrito mais à frente. Assim, existe na plataforma uma classe específica que tem a responsabilidade de gerir todos os componentes constituintes qualquer modelo implementado. Esta classe é constituída essencialmente por dois métodos:

- LoadPlugin
- Execute

Método "LoadPLugin"

O método "LoadPlugin" utiliza o ficheiro de configuração geral da plataforma, denominado "ABM.xml" para localizar o ficheiro de configuração dos componentes. O ficheiro "ABM.xml" é um ficheiro do tipo XML que tem o objectivo de registar todos os parâmetros de configuração da plataforma que tenham um âmbito geral e global: o nome do ficheiro de configuração dos componentes para um dado modelo é um bom exemplo de um parâmetro de configuração deste tipo.

Através da leitura do ficheiro de configuração dos componentes, a plataforma identifica as características de cada componente constituinte os vários modelos configurados. Mais concretamente, por cada modelo configurado, a plataforma identifica o nome da classe que implementa o componente e o nome do módulo em que esta classe se encontra implementada.

Esta informação é colocada numa lista que sendo indexada por nome de modelo e nome de componente, contém todos os componentes. Assim, através desta lista, a plataforma consegue localizar o componente para um dado modelo que corresponde a uma determinada fase do ciclo principal de execução.

Método "Execute"

O método "Execute" é muito simples dado que, através da lista referida no ponto anterior, basta saber qual é o nome do modelo que está a ser executado e o nome da fase do ciclo principal de execução que se pretende executar.

6.2.3. Classes Nucleares

O terceiro pilar fundamental da plataforma aplicacional é o conjunto de classes que implementam os conceitos fundamentais do quadro de referência conceptual subjacente à plataforma, tendo-se dado a este conjunto de classes o nome de Classes Nucleares.

Há que considerar que estas classes implementam as propriedades e o comportamento genérico das principais entidades que caracterizam em geral um Modelo Baseado em Agentes e que num dado modelo específico, será geralmente necessário definir classes derivadas destas que implementam as características específicas do modelo em questão.

As classes que pertencem ao conjunto das Classes Nucleares são:

- Classe "Thing";
- Classe "Agent";
- Classe "Morphology";

- Classe "Population";
- Classe "Environment";
- Classe "Grid";
- Classe "Cell";
- Classe "World";

Em seguida são apresentados os aspectos principais de cada uma destas classes, referindo as propriedades, métodos e construtores mais relevantes, e evidenciando como se enquadram no quadro de referência conceptual.

Classe "*Thing*"

Classe "*Thing*"	
Propriedades	– `IEnvironment Environment` – `ICell Cell`
Métodos	– `void Move(int x, int y, bool relativeFlag)`
Construtores	– `Thing(ICell cell)`

Tabela 1 – Propriedades, Métodos e Construtores da Classe "Thing"

A classe "*Thing*" não tem uma correspondência directa no Quadro de Referência Conceptual e é a classe-mãe de todas as classes que poderão ter um posicionamento espacial. Na versão actual, somente a classe "*Agent*" deriva desta classe.

A definição e utilização desta classe resultam numa implementação da plataforma mais simples e mais organizada. Nesse sentido, a classe "Thing" tem duas propriedades (a localização espacial através da referência à célula da matriz espacial, "Cell", e a referência ao objecto que representa o ambiente do modelo, "Environment") e o método "Move" que implementa o comportamento de deslocação (tanto relativa como absoluta) na matriz. Por outro lado, a classe "Thing" tem um construtor em que são inicializadas as propriedades "Cell" e "Environment".

Na versão actual da plataforma, somente a classe "Agent" deriva da classe "Thing"; no entanto, poder-se-ão definir, em versões futuras, outras entidades derivadas da classe "Thing", como, por exemplo, uma entidade que represente obstáculos na matriz espacial.

Classe "*Agent*"

Classe "*Agent*"	
Propriedades	– IPopulation Population – IMorphology Morphology – int Index
Métodos	– void Move(ICell cellToMoveTo)
Construtores	– Agent(IPopulation population, int index) – Agent(IPopulation population, int index, BitArray vector) – Agent(IPopulation population, int index, int value, ICell gridCell) – Agent(IPopulation population, ArrayList morphology, ICell gridCell)

Tabela 2 – Propriedades, Métodos e Construtores da Classe "*Agent*"

Esta classe é a representação mais genérica da entidade Agente, sendo a única classe que deriva da classe "*Thing*".

Propriedades

As principais propriedades desta classe representam a população à qual o agente pertence e a morfologia do agente. Ambas estas propriedades são representadas por outras duas classes nucleares da plataforma descritas mais adiante (respectivamente a classe "*Population*" e a classe "*Morphology*").

Métodos

O único método desta classe é o método "Move" que representa a deslocação do agente na matriz ("Grid") em que se encontra. É um método utilizado, portanto, nos modelos em que existe uma representação espacial da localização dos agentes e em que é relevante a possibilidade se poder deslocar no espaço em que se encontra.

Construtores

A classe "Agent" tem vários construtores de modo a contemplar vários tipos de agentes. Essencialmente, contemplam-se agentes com posicionamento espacial ou não, em que, quando se cria o agente se especifica também a sua localização, e agentes com vários tipos de morfologia (escalar, vectorial e matricial) conforme se descreverá mais adiante quando se apresentar a classe "Morphology".

Classe "Morphology"

Classe "*Morphology*"	
Propriedades	– IAgent Agent – double Value – int VectorLength – int[] ValuesVector – int[] MatrixSize – ArrayList Matrix
Métodos	– void ChangeBit(int index) – void ChangeVector(int vectorIndex, BitArray newVector)
Construtores	– Morphology(IAgent agent) – Morphology(IAgent agent, int value) – Morphology(IAgent agent, ArrayList matrix)

Tabela 3 – Propriedades, Métodos e Construtores da Classe "*Morphology*"

Esta classe representa a forma como o agente é percepcionado no âmbito do modelo, ou seja, representa a forma do agente. Na versão actual da plataforma, foram contempladas três tipos de forma: escalar, vectorial e matricial. No primeiro caso, a forma do agente é representada por um único valor; no segundo caso, por um vector de bits; no terceiro caso, por um conjunto de vectores de bits com a mesma dimensão.

É fácil de compreender que o tipo de representação da forma com que o agente se apresenta aos outros agentes tem a ver essencialmente com o fenómeno que se deseja estudar e como se considera mais apropriado modelizar o funcionamento dinâmico do modelo que representa o modelo. Por exemplo, como veremos mais adiante, no capítulo em que apresentam os vários modelos específicos que foram implementados com a plataforma, no modelo relativo à Dinâmica de Opiniões, optou-se por uma morfologia do tipo vectorial e no modelo relativo à Disseminação da Cultura, optou-se pelo tipo matricial.

Propriedades

As propriedades da classe "Morphology", para além da referência ao agente ao qual a morfologia pertence, contêm os objectos necessários à representação dos vários tipos de morfologia possível (escalar, vectorial e matricial). Por exemplo, a dimensão da matriz e os valores da matriz para o caso de uma morfologia matricial.

Métodos

Existem dois métodos que implementam a alteração de valores de morfologias do tipo vectorial e do tipo matricial: respectivamente os métodos "ChangeBit" e "ChangeVector".

Construtores

Os construtores contemplam também os vários tipos de morfologias possíveis.

Classe "Population"

Classe "*Population*"	
Propriedades	– `IWorld World` – `int MaxGenerationsNumber` – `int CurrentGeneration` – `int AgentsNumber` – `double AgentMaxValue` – `List<IAgent> AgentsList` – `IAgentsGroups AgentsGroups` – `int[,] InteractionMatrix`
Métodos	– `void Init(List<IAgent> agentsList)`
Construtores	– `Population(IWorld world, int agentsNumber)`

Tabela 4 – Propriedades, Métodos e Construtores da Classe "*Population*"

Esta classe agrupa todos os agentes que constituem o modelo. Na versão actual, a plataforma não prevê a existência de mais do que uma população por modelo, porque os modelos implementados não tinham essa exigência; no entanto, a generalização para modelos com mais de uma população, agrupando cada população os agentes do mesmo tipo, é facilmente implementada.

Propriedades

A classe "Population" contém várias propriedades:

- Referência ao mundo ao qual a população pertence (ver a descrição da classe "World" mais adiante);
- O número máximo de gerações que a população poderá ter;
- Um identificador da geração actual;
- O número de agentes que constituem a população;
- O valor máximo que os agentes poderão ter (aplicável somente nos casos em que os agentes tem uma morfologia do tipo escalar);
- A lista dos agentes que constituem a população;
- A lista dos grupos de agentes agrupados segundo um dado critério característico do modelo específico (existe uma classe específica para os grupos de agentes, denominada "AgentsGroups" que implementa uma lista de grupos e que disponibiliza o método de adição de uma agente a um grupo e de remoção de um agente de um grupo);

- A matriz de interacção entre agentes (nesta matriz é guardado o número de interacções entre todos os agentes que constituem o modelo, tomados dois a dois).

Métodos

A classe "Population" só tem um método: o método "Init" é chamado na altura em que o modelo é inicializado (o mundo é criado) e tem o objectivo de inicializar a população com todos os agentes previstos para o modelo.

Construtores

A classe "Population" só tem um construtor onde são criadas e inicializadas as várias propriedades da classe com base nos valores definidos para o modelo específico. Foi criado um método separado e específico para a inicialização da lista de agentes (método "Init" referido no ponto anterior) por conveniência de implementação.

Classe "Environment"

Classe "*Environment*"	
Propriedades	– IWorld World – IGrid Grid
Métodos	
Construtores	– Environment(IWorld world – Environment(IWorld world, int[,] grid)

Tabela 5 – Propriedades, Métodos e Construtores da Classe "*Environment*"

A classe "Environment" representa o ambiente em que os agentes actuam. É uma classe muito simples e genérica que só contém duas propriedades e dois construtores.

Propriedades

As únicas propriedades que a classe "Environment" contém são as referências aos objectos que representam o mundo e a matriz espacial do modelo.

Métodos

A classe "Environment" não contém métodos.

Construtores

Os dois construtores existentes prevêem a situação do modelo ter, ou não, uma matriz espacial ("Grid").

Classe "Grid"

Classe "*Grid*"	
Propriedades	– IEnvironment Environment – int XDimension – int YDimension – ICell[,] Cells – List<ICell> VacantCellsList
Métodos	
Construtores	– Grid(IEnvironment environment, int xDim, int yDim)

Tabela 6 – Propriedades, Métodos e Construtores da Classe "*Grid*"

Na versão actual da plataforma, o posicionamento espacial dos agentes, nos modelos em que tenha sentido representar esse posicionamento, é efectuado assumindo uma matriz bidimensional constituída, por um número de linhas e um número de colunas prefixados, e assumindo que o cruzamento entre uma linha e uma coluna da matriz corresponde a uma célula representada pela classe "Cell".

A classe "Grid" descreve esta matriz espacial e só terá aplicação nos casos em que o posicionamento espacial dos agentes, em sentido absoluto ou relativo, tenha relevância para o estudo do modelo em estudo.

Propriedades

A classe "Grid" contém as seguintes propriedades:

- Uma referência ao objecto que representa o ambiente do modelo;
- As duas dimensões da matriz;
- Uma lista bidimensional das células que constituem a matriz;
- Uma lista das células vazias (células que não contêm agentes).

Métodos

A classe "Grid"não contém métodos.

Construtores

A classe "Grid" contém um construtor em que são inicializadas as propriedades e em que são criadas as células que constituem a matriz espacial.

Classe "Cell"

Classe "*Cell*"	
Propriedades	– IGrid Grid – IThing thing – IPosition Position – ICell[] Neighbors – Dictionary<double, double> NeighborsWeigths
Métodos	– void GetNeighbors(– void UpdateNeighborsWeigths()
Construtores	– Cell(Grid grid, int x, int y)

Tabela 7 – Propriedades, Métodos e Construtores da Classe "*Cell*"

A classe "Cell" representa qualquer posição possível da matriz espacial de posicionamento dos agentes.

Propriedades

As propriedades da classe "Cell" são:

- Referência à matriz à qual a célula pertence;
- Referência ao objecto que a célula eventualmente contém (esta referência é efectuada através de um objecto do tipo "Thing" já descrito acima;
- Posição espacial da célula através das duas coordenadas do plano da matriz;

- Lista das oito células adjacentes à célula em questão (a matriz espacial é considerada toroidal no sentido que as células localizadas nas linhas e colunas limítrofes da matriz terão algumas células adjacentes localizadas nas linhas e colunas diametralmente opostas);
- Lista dos pesos relativos das células adjacentes considerando as percentagens de células contendo um agente com um dado valor em relação ao número total de células adjacentes não vazias.

Métodos

Os métodos da classe "Cell" são dois:

- O método "GetNeighbors" que actualiza a lista das células adjacentes;
- O método "UpdateNeighborsWeigths" que actualiza os pesos relativos das células adjacentes.

Construtores

A classe "Cell" só possui um construtor onde são inicializadas todas as propriedades de cada célula da matriz à matriz que forem sendo instanciadas.

Classe "World"

Classe "*World*"	
Propriedades	– `string` ModelName – `Random` Random – `IEnvironment` Environment – `IPopulation` Population – `Form` Form – `bool` DoNothing – `bool` DoNextCycle
Métodos	
Construtores	– World(`string` modelName, `Form` form, `int` seed

Tabela 8 – Propriedades, Métodos e Construtores da Classe "*World*"

A classe "World" pretende representar a simulação na sua globalidade: considera-se que caracteriza o "mundo" em que corre o modelo. Tudo o que tenha a ver com o modelo deve poder ser acedido a partir da classe "World".

Propriedades

As propriedades da classe "World" correspondem às variáveis genéricas que caracterizam o modelo na sua globalidade e que são utilizadas ao longo da simulação:

- O nome do modelo é utilizado para localizar os componentes a utilizar no ciclo de execução do modelo conforme anteriormente descrito no ponto relativo ao gestor de componentes;
- Uma referência ao objecto que determina a sequência de valores aleatórios a aplicar durante a simulação;

- Uma referência ao objecto que contém o ambiente do modelo;
- Uma referência ao objecto que contém a população do modelo;
- Uma referência ao objecto que implementa a interface de utilizador do modelo;
- Uma variável que indica se o modelo está ou não em modo suspenso por vontade do utilizador;
- Uma variável que indica se o ciclo o ciclo de execução deve ser concluído.

Métodos

Esta classe não contém métodos.

Construtores

A classe "World" só contém um construtor em que algumas das propriedades são inicializadas com os dados específicos do modelo: em particular, o objecto que define a sequência de valores aleatórios é inicializado com uma semente, semente esta definida ou pelo utilizador ou de forma aleatória.

6.3. Arquitectura do Modelo Específico

Em relação à arquitectura de um dado modelo específico, há que considerar três entidades principais:

- Interface de Utilização;
- Classes Específicas;
- Componentes.

A Interface de Utilização só deve ser implementada no âmbito do modelo específico, devendo ficar completamente fora da plataforma aplicacional. Este requisito fundamental garante a possibilidade de ter vários tipos de interface com o utilizador, por exemplo, para ecrã ou para ficheiro, evitando simultaneamente que esta versatilidade torne a plataforma mais complexa.

As Classes Específicas são constituídas pelas classes derivadas das classes fundamentais da plataforma e que implementam as propriedades e os métodos específicos que caracterizam as entidades do modelo específico. Por exemplo, num dado modelo, a entidade agente poderá ter que ter uma propriedade adicional ou ter um método com uma implementação diferente da que já está prevista na plataforma.

Os Componentes de um dado modelo implementam as funções primordiais que são evocadas pelo motor da plataforma no ciclo de execução do modelo e que já foram referidos na descrição da arquitectura da plataforma.

6.3.1. Interface de Utilização

A Interface de Utilização tem a responsabilidade de gerir os ecrãs de recolha das variáveis de entrada do modelo e de apresentar os resultados das simulações efectuadas. Permite também que o utilizador suspenda e retome a execução do modelo quando achar oportuno.

Deve-se também considerar contemplados no âmbito da interface de utilização a eventual actualização de ficheiros com os resultados para posterior análise ou de ficheiros de "Log" com o registo das acções realizadas.

6.3.2. Classes específicas

No âmbito do modelo específico, existem muitas vezes especificidades dos conceitos gerias do quadro de referência que exigem uma implementação específica. Nesse sentido, a utilização de uma linguagem orientada por objectos e a arquitectura da plataforma aplicacional facilitam notavelmente esta tarefa: são criadas ao nível do modelo classes específicas derivadas das classes definidas na plataforma que implementam as particularidades requeridas. Esta situação ocorreu em alguns dos modelos de exemplo que foram criados.

6.3.3. Componentes

De acordo com o dito na descrição da plataforma aplicacional, os componentes que são definidos ao nível do modelo específico implementam os passos do ciclo de execução que devem ser contemplados considerando os requisitos funcionais do modelo em questão.

Para que um dado componente possa ser assim executado pelo motor aplicacional, tem que estar de acordo com uma arquitectura prefixada, implementando a interface preestabelecida.

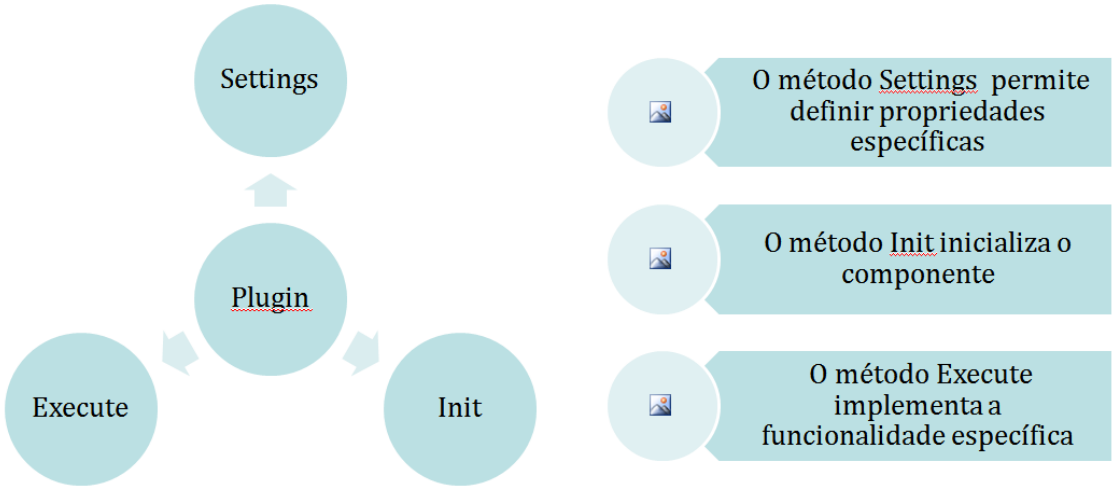

Figura 5 – A Arquitectura do Componente

A arquitectura do componente é constituída por uma propriedade e por dois métodos.

A propriedade "Setttings" implementa uma lista genérica que pode ser utilizada para conter um número variável de propriedades específicas do componente.

O método "Init" é chamado pelo Gestor de Componentes no memento em que o componente é carregado conforme referido na descrição da plataforma aplicacional.

O método "Execute" é o método chamado pelo motor da plataforma aplicacional e contém a funcionalidade específica que deve ser executado no passo considerado do ciclo de execução do modelo.

6.4. Aspectos genéricos

Existem dois aspectos relacionados com implementação da plataforma aplicacional que merecem ser referidos e, como não se podem considerar opções fundamentais sob o ponto de vista da arquitectura da aplicação, são apresentados separadamente neste ponto.

Estes aspectos são:

- Sequências aleatórias;
- Execução simultânea.

6.4.1. Sequências aleatórias

Muito frequentemente, quando se executam simulações é necessário recorrer, em vários momentos ao longo da simulação, a valores aleatórios que reproduzam o mais fielmente possível a existência de casualidade no fenómeno estudado.

No entanto, por outro lado, há sempre a necessidade de garantir que uma simulação executada num dado modelo possa ser integralmente reproduzida em qualquer momento. Esta necessidade deriva fundamentalmente do facto que os modelos criados devem poder ser sempre validados e uma das técnicas de validação de um modelo é a verificação de todas os passos executados pelo modelo com um nível de detalhe máximo.

Ora se o modelo recorre a valores aleatórios nos vários passos que constituem a simulação, como geralmente acontece, é fácil de entender que neste caso só será possível reproduzir integralmente a simulação se se conseguir reproduzir exactamente a sequência de valores aleatórios utilizada.

A utilização das chamadas sequências "pseudo-aleatórias" na plataforma aplicacional criada permite vir ao encontro desta necessidade. Numa sequência "pseudo-aleatória", existe um valor que determina a sequência de valores aleatórios, denominado semente, no sentido que, para uma dada semente, os vários valores obtidos na sequência são sempre os mesmos, apresentando esta sequência, no entanto, um comportamento estatístico aleatório.

Assim, como se pode ver nos modelos exemplo implementados, com a plataforma aplicacional criada, o utilizador tem duas opções na interface de utilização: ou escolhe explicitamente o valor da semente a utilizar ou, em alternativa, deixa que seja a plataforma a atribuir um valor à semente da sequência aleatória. Neste último caso, a semente é calculada com base no instante em que é calculada, calculado com uma resolução de 100 nano segundos com base no relógio interno do computador.

Desta forma, será sempre possível correr um modelo com uma sequência de valores aleatórios sempre diferente ou, se se quiser, com a mesma sequência de valores aleatórios.

6.4.2. Execução simultânea

A arquitectura da plataforma aplicacional escolhida e a forma como foi implementada permitem executar um número ilimitado de modelos diferentes em simultâneo (os termos "ilimitado" e "simultâneo" devem ser interpretados considerando que a execução dos modelos é realizada num sistema computacional com capacidade de processamento limitada).

Nesse sentido, foi criado um programa muito simples para lançamento dos modelos criados: apresenta uma lista dos modelos disponíveis e o utilizador pode escolher o modelo que deseja executar. O modelo é executado num processo autónomo de modo a que o utilizador possa lançar de imediato outro modelo, e assim sucessivamente.

Portanto a chave para se conseguir a execução simultânea dos modelos é lançar a sua execução em processos separados do sistema operativo, funcionalidade imediatamente disponível no ambiente .Net escolhido para o desenvolvimento da plataforma aplicacional em questão.

Seria, até possível, com toda a facilidade, alterar a interface de utilização do programa de lançamento, de modo a permitir a selecção de vários modelos e depois o lançamento de todos os modelos em conjunto.

7. Exemplos de aplicação

Para comprovar que a plataforma aplicacional criada é adequada à construção de Modelos Baseados em Agentes de vários tipos foram implementados quatro modelos com características diferentes:

- Culture Dissemination Model;
- El Farol Bar Problem;
- Opinion Dynamics Model;
- Segregation Model.

Estes modelos são modelos que foram criados para o estudo de fenómenos sociais concretos e para a verifição de hipóteses teóricas concebidas por vários investigadores que se dedicaram ao estudo destes fenómenos, existindo uma extensa bibliografia que descreve a estrutura destes modelos, a sua finalidade e os fundamentos teóricos.

A aplicação da pltaforma aplicacional à criação dos modelos acima referidos não teve como único objectivo a comprovação da adequabilidade da plataforma: foi também muito útil para identificar funcionalidades que deveriam ser contempladas pela plataforma.

Por cada modelo, a descrição abaixo apresentada é constituída por duas partes: na primeira parte, é descrito o fenómeno que se pretende estudar e como foi concebido o respectivo modelo; na segunda parte, são referidos os principais aspectos da implementação do modelo sobre a plataforma aplicacional objecto do presente trabalho.

Como foi já referido, a implementação de um modelo específico sobre a platfaorma aplicacional contempla essencialmente três aspectos:

- A criação dos componentes correspondentes aos passos do ciclo de execução que no modelo em questão deverão ser considerados;
- A criação das classes específicas ao modelo derivadas das classes nucleares da plataforma;
- A criação da interface de utilização para o modelo específico para recolha dos parâmetros de configuração e para apresentação dos resultados;

Serão precisamente estes os aspectos que serão referidos na descrição da implementação de cada modelo.

7.1. O Modelo de Disseminação de Culturas

7.1.1. Enquadramento teórico

Em 1997, Robert Axelrod publicou um artigo com o título "The Dissemination of Culture: A Model with Local Convergence and Global Polarization" onde apresentou uma teoria para responder a seguinte questão: porque é que, havendo uma maior tendência para as pessoas assumirem as mesmas convicções e atitudes, como consequência das interacções entre si, as diferenças não acabam por desaparecer?

Ou seja, como se pode explicar o facto que continuam a existir regiões com culturas diferentes apesar de haver uma grande tendência para a uniformização da cultura provocada pela expansão e penetração dos meios de comunicação?

Ficou célebre a frase "Local convergence can lead to global polarization" que sintetiza o resultado deste estudo.

Os estudos sobre como os processos dinâmicos de influência social se reflectem nas transformações culturais de grupos sociais utilizam frequentemente Modelos Baseados em Agentes constituídos por um certo número de regiões geográficas, em que a cultura de cada região é representada por uma matriz constituída por um conjunto de vectores de bits, onde cada vector caracteriza uma das dimensões da cultura (aspecto cultural), podendo o vector assumir os vários valores diferentes que essa dimensão permite (traço do aspecto cultural) - ou seja o traço cultural de uma região (ou grupo de indivíduos ou até indivíduo) deve ser entendido como um dos possíveis valores que um determinado aspecto cultural dessa região (ou do grupo de indivíduos ou do indivíduo) pode assumir. Esta caracterização da cultura, decompondo-a em aspectos e traços e representando-a matricialmente, permite atribuir às culturas de duas regiões um coeficiente de afinidade calculado como a percentagem de aspectos culturais que têm o mesmo traço.

A utilização destes Modelos Baseado em Agentes, em que se assume também que, quanto mais a cultura de uma região for similar à cultura das regiões adjacentes, maior é a probabilidade que a cultura dessa região se tornar ainda mais semelhantes à cultura das regiões adjacentes, permite chegar a algumas conclusões muito significativas, como, por exemplo: o número de regiões com a mesma cultura diminui com o número de dimensões culturais consideradas, aumenta com o número possível de valores diferentes que cada dimensão pode ter e, a partir de um certo número de regiões iniciais, diminui com o aumentar desse número.

7.1.2. Implementação do modelo

Interface de Utilização

A interface de utilização deste modelo apresenta um ecrã com três áreas distintas:

- Uma área contém os parâmetros de configuração do modelo;
- Outra área mostra o resultado da simulação como um mapa de regiões com a respectiva cultura representada com uma cor e apresenta uma tabela indicando o número de regiões com a mesma cultura e o valor da respectiva cultura;
- Por fim, uma terceira área onde é apresentada uma tabela com os resultados dos várias simulações.

Os parâmetros de configuração do modelo podem-se subdividir em dois grupos:

- Parâmetros que caracterizam o universo que o modelo representa;
- Parâmetros que definem a forma como o modelo é apresenta do.

No primeiro grupo encontrem-se os seguintes parâmetros:

- Número de agentes, ou seja, neste contexto, número de regiões;
- Número de dimensões culturais;
- Número de traços por dimensão cultural;
- Número máximo de ciclos de execução;
- Semente de sequência aleatória.

O segundo grupo de parâmetros é constituído por:

- Dimensão do lado do quadrado que representa a cultura de uma região;
- Velocidade de execução determinada pelo tempo de espera especificado.

A tabela com o resultado das simulações efectuadas contém, por simulação, o número de agentes, o número de dimensões culturais, o número de valores possíveis por dimensão, o número máximo de ciclos de execução e o número de grupos formados.

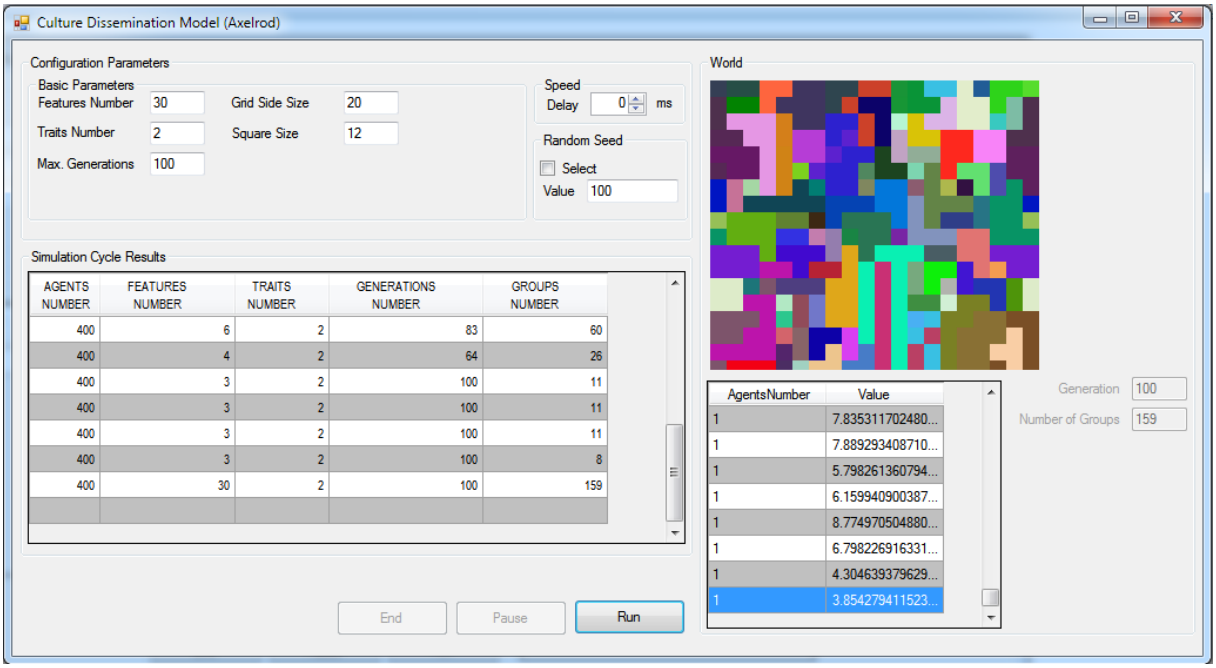

Figura 6 – O Modelo de Disseminação de Culturas

Classes Específicas

Neste modelo não foi necessário criar classes específicas.

Componentes

Os componentes construídos neste modelo foram:

- "InitWorld";
- "SocializePopulation";
- "FinalizeWorld";
- "ShowWorld".

No componente "InitWorld", são criados o ambiente e a população do modelo. Seguidamente, a população é inicializada com agentes cuja cultura é atribuída aleatoriamente de acordo com os parâmetros de configuração do modelo. Por fim, é criada a lista com os vários grupos de agentes que têm a mesma cultura.

O componente "SocializePopulation" executa um ciclo em que por cada agente do modelo, identifica o parceiro (agente adjacente com maior grau de afinidade cultural) e altera a primeira dimensão cultural do agente que seja diferente da correspondente dimensão do parceiro de modo a ficar igual. Assim a cultura de cada agente ficará mais semelhante à cultura do agente adjacente mais afim.

O componente FinalizeWorld actualiza simplesmente a tabela com os resultados da simulação.

Por fim, o componente ShowWorld mostra o mapa das regiões com uma cor diferente por cada valor da respectiva cultura de modo a se poder visualizar, o mais facilmente possível, as regiões com a mesma cultura.

7.2. O Jogo Minoritário "El Farol Bar"

7.2.1. Enquadramento teórico

O problema vulgarmente conhecido como problema de "El Farol" foi criado por Brian Arthur, em 1994, para investigar a melhor forma de modelizar o comportamento dos agentes económicos caracterizados por terem uma racionalidade limitada.

"El Farol" é o nome de um bar em Santa Fé, nos Estados Unidos, que, em todas as quintas-feiras à noite, tem um espectáculo de música. Brian Arthur inspirou-se nesta situação para criar o seguinte problema: Considere-se que existe um número total de possíveis participantes no evento do bar "El Farol", e assuma-se que qualquer um desses possíveis participantes só ficará satisfeito com a ida ao evento se o número das pessoas efectivamente presentes é inferior a 60% do número total dos possíveis participantes. Nenhum dos participantes, ao tomar a decisão se ir o não ao evento, sabe qual é a decisão que os outros participantes tomarão e a única informação que tem e que poderá ser utilizada para tomar a decisão de ir, ou não, é a efectiva percentagem de participantes nos eventos anteriores.

Arthur mostrou, com um Modelo Baseado em Agentes, que o número de participantes no evento converge rapidamente, para depois oscilar ciclicamente em torno da percentagem limite (60%).

É óbvio que se existisse um único método para a tomada de decisão, então todos os possíveis participantes utilizariam esse método, chegariam a mesma decisão e todos juntos acabariam por ir, ou alternadamente não ir, ao evento: ou seja, todos ficariam sempre insatisfeitos. É comum este tipo de situações em fenómenos económicos ou sociais, em que um agente económico ou social tem que tomar uma estratégia fundamentando-a nas decisões que o agente assume que os outros agentes intervenientes tomarão. Ou seja, nestas situações, os

agentes intervenientes não se podem considerar como total e perfeitamente racionais: os agentes apresentam, pelo contrário, racionalidade limitada e tomam as suas decisões em previsões efectuadas baseadas nos padrões observados, resultantes da experiência realizada.

No Modelo Baseado em Agentes criado para o problema de "El Farol", é exactamente assim que cada agente procede: utiliza um dos algoritmos do seu repositório para prever a participação no próximo evento com base nas participações efectivas nos eventos passados, escolhendo o algoritmo que apresenta uma maior taxa de sucesso. Obviamente que este modelo de previsão exige que o agente actualize sempre a taxa de sucesso dos vários algoritmos que pode utilizar considerando a taxa de participação efectiva em cada evento realizado.

7.2.2. Implementação do modelo

Interface de Utilização

A interface de utilização do modelo é constituída fundamentalmente por duas áreas:

- Uma área contém os parâmetros de configuração do modelo;
- Outra área contém o resultado da execução do modelo.

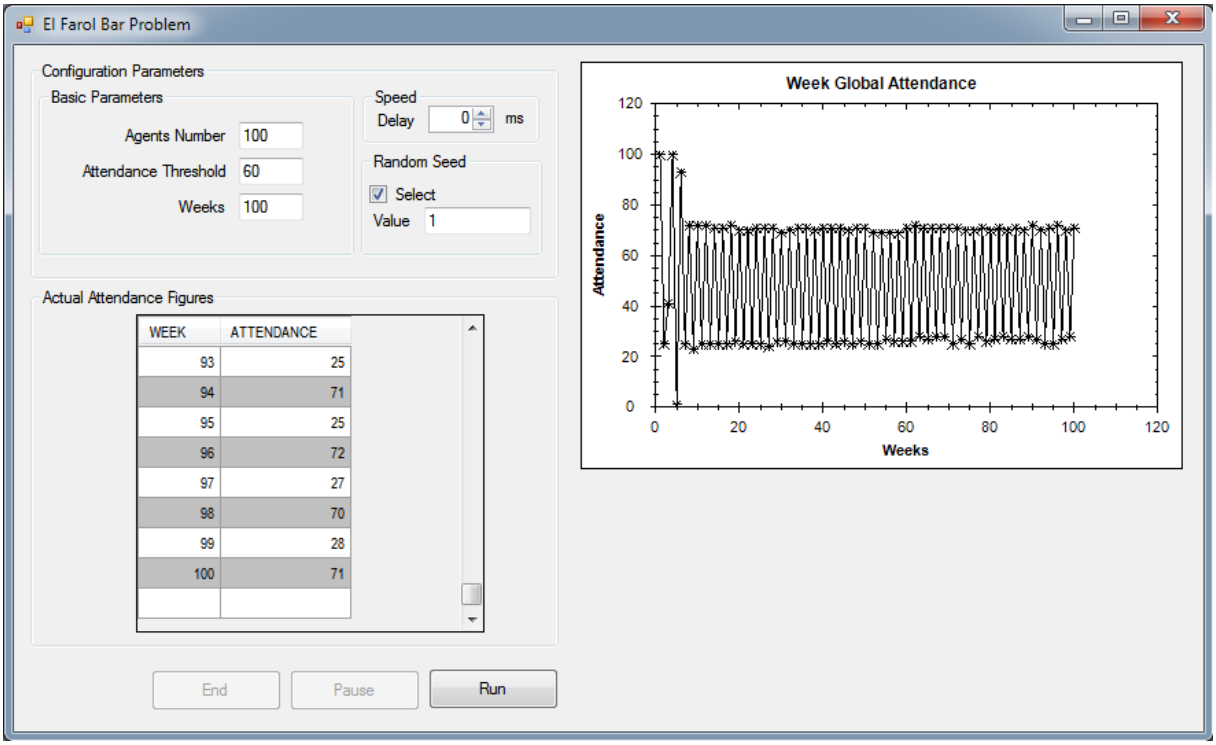

Figura 7 – O Modelo "El Farol Bar"

Os parâmetros de configuração do modelo são:

- Número de agentes;
- Limiar de participação (percentagem de participantes a partir da qual a satisfação se torna negativa);
- Número de eventos a simular;
- Semente da sequência aleatória;
- Velocidade de execução do modelo determinada pelo intervalo de pausa cumprido em cada ciclo de execução.

O resultado da execução do modelo é representado por um diagrama e por uma tabela que mostram a percentagem de participação ao longo das várias semanas.

Em termos de controlo de execução do modelo, estão disponíveis três botões:

- Um botão que lança a execução de um ciclo de simulações;
- Outro botão que suspende a execução da simulação em curso;
- E, por fim, um botão que termina de imediato a execução.

Classes Específicas

Neste modelo só foi necessário criar a classe específica "Agent" derivada da classe "Agent" definida na plataforma aplicacional.

Na classe específica "Agent" foram acrescentadas duas propriedades:

- "Predictors";
- "Forecast".

A propriedade "Predictors" é uma lista de objectos que identificam os vários algoritmos de previsão que o agente em questão poderá utilizar para determinar a percentagem de participação que prevê para o próximo evento. Cada algoritmo é identificado por um objecto do tipo "PredictorInfo" que contém três campos:

- Um apontador para a função que implementa o algoritmo;
- Um campo contendo a previsão calculada com base na aplicação do algoritmo correspondente;

- Um campo contendo a precisão do algoritmo, precisão calculada com base nos valores efectivos de participação nos eventos já realizados.

A propriedade "Forecast", ao nível da classe "Agent", contém a previsão que o agente obtém aplicando o algoritmo de previsão escolhido (o que apresenta maior precisão).

E foram, também, acrescentados, à classe "Agent", um novo construtor e um novo método ("Predict").

O construtor estabelece, aleatoriamente, o número de algoritmos de previsão a considerar para o agente em questão e identifica, no conjunto de todos os algoritmos previstos, os que se devem considerar associados ao agente em questão.

O método "Predict" calcula a previsão considerando o algoritmo de previsão com maior precisão ou, caso todos os algoritmos tenham uma precisão nula (facto que ocorre na primeira semana), escolhe aleatoriamente o algoritmo a aplicar.

Componentes

Neste modelo foram criados quatro componentes:

- "InitWorld";
- "SeekTarget";
- "FinalizeWorld";
- "ShowWorld".

No componente "InitWorld" é criada e inicializada a população constituída pelos agentes que compõem o modelo.

O componente "SeekTarget" calcula, por cada agente do modelo, a previsão de participação no próximo evento, utilizando o método "Predict" da classe "Agent".

No componente "FinalizeWorld" não é efectuada nenhuma acção relevante a não ser inibir e activar os campos de input e de output da simulação de modo a manter a consistência da interface de utilização.

Por fim, o componente "ShowWorld" calcula o efectivo índice de participação e, com base nesse valor, actualiza a precisão dos vários algoritmos utilizados por cada agente.

7.3. Um Modelo de Dinâmica de Opiniões

7.3.1. Enquadramento teórico

Com base em um estudo sobre os processos de troca de opinião em grupos sociais (Banisch et al, 2009) foi construído um modelo sobre a plataforma aplicacional objecto do presente trabalho. No âmbito deste estudo, a opinião de um qualquer agente do grupo social é representado por um vector constituído por uma sequência de bits com uma dimensão prefixada e comum. Considera-se que a interacção entre dois agentes só poderá ocorrer se o número de bits diferentes nos respectivos vectores de opinião for inferior ou igual a um dado valor prefixado, ou seja se o nível de afinidade entre os dois agentes é superior a um limite prefixado, consistindo esta interacção na mudança de um dos bits diferentes do vector de opiniões de um dos dois agentes de forma a aumentar a afinidade entre os dois agentes. Ou seja, se os dois agentes forem suficientemente afins, ficarão ainda mais afins; se não forem suficiente afins, não se relacionam e ficam exactamente como estão.

O estudo analisa o impacto dos vários parâmetros característicos do fenómeno (e obviamente também do modelo criado para o efeito) nos resultados finais da simulação, como, por exemplo, a dimensão do vector representando a opinião do agente, o limiar de afinidade, a dimensão do grupo social, etc., e preocupa-se fundamentalmente com a análise da fase de transição entre uma morfologia do grupo caracterizada pela grande fragmentação das opiniões e uma morfologia caracterizada pela homogeneidade de opiniões. Por fim, a análise e classificação das redes de comunicações resultantes das interacções ocorridas entre os vários agentes do grupo é efectuada tendo em vista a identificação de padrões emergentes e estruturas sociais resultantes dum processo de comunicação simples.

Um dos principais resultados deste estudo foi confirmar o que outros estudos anteriores também apresentaram: a troca de opiniões num grupo social no sentido de tornar os agentes mais afins entre si não comporta forçosamente a uniformidade de opiniões em todo o grupo social.

A execução do modelo é constituída fundamentalmente pelos seguintes passos:

1) Um prefixado número de agentes é criado, tendo cada um, uma matriz de opiniões (representadas por vectores de bit com uma dimensão prefixada) criadas aleatoriamente;

2) Em cada ciclo, são escolhidos pares de agentes aleatórios, são comparadas as respectivas opiniões entre si e, se estas diferirem em menos de um número prefixado de bits (coeficiente de afinidade), é alterado um dos bits diferentes do vector opinião de um dos agentes, aumentando a afinidade entre os dois agentes (o número de comparações realizadas em cada ciclo é igual ao número de agentes do grupo social, mas como os agentes são escolhidos aleatoriamente não existe a garantia que todos os agentes intervenham nestas comparações);

3) É executado um número de ciclos igual a um valor prefixado ou são executados quantos ciclos forem necessários para que o modelo chegue a uma situação em que já não é possível uma mudança de opinião em nenhum agente mesmo considerando uma possível interacção com qualquer outro agente, isto é, quando se chegar a um estado em que os grupos de opiniões criados não são afins entre si.

7.3.2. Implementação do modelo

Interface de Utilização

Neste modelo, a interface de utilização é constituída por um ecrã em que se encontram três áreas principais definidas:

- Uma área contém os vários parâmetros de configuração;
- Outra área contém a visualização dos vários agentes com as respectivas opiniões durante a execução do ciclo de simulação mostrando como as opiniões vão alterando e apresenta a lista dos grupos formados;
- Por fim, numa última área é mostrada uma tabela com os resultados de todas as simulações realizadas.

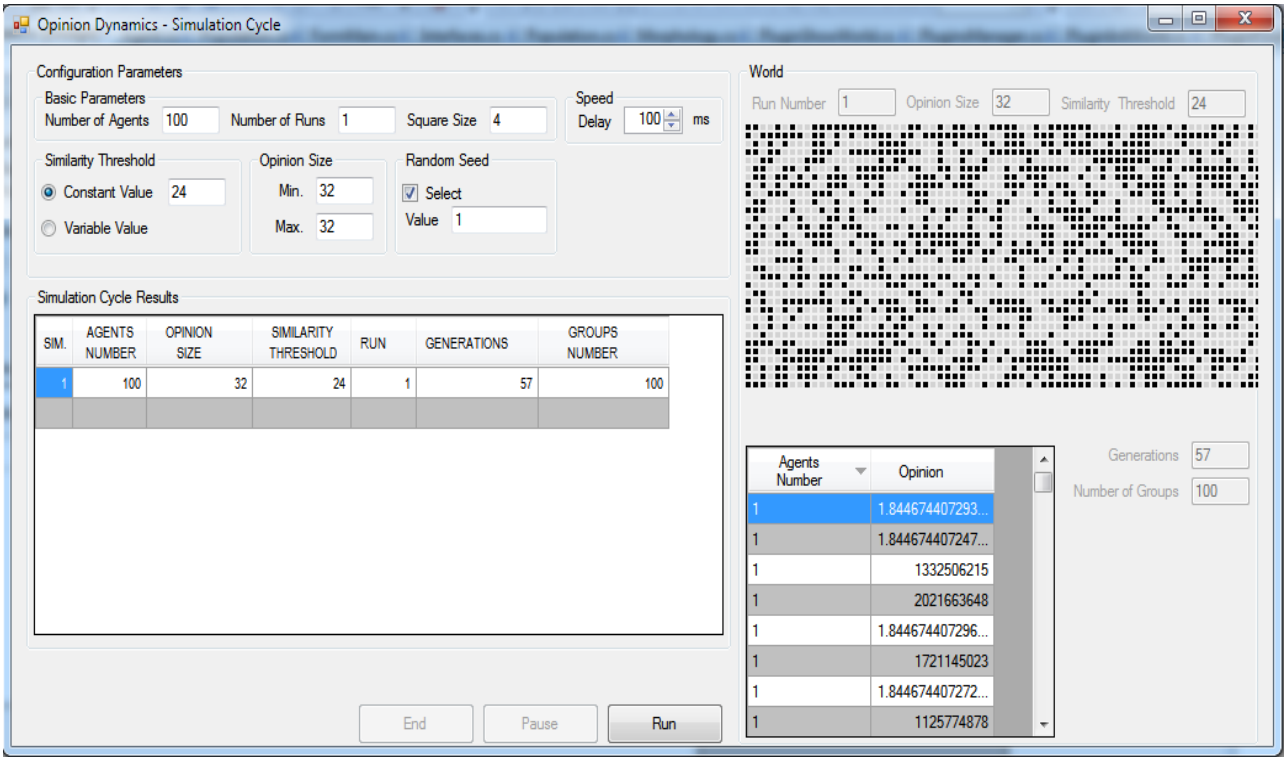

Figura 8 – O Modelo "Dinâmica de Opiniões"

Os parâmetros principais que caracterizam a configuração do modelo são:

- Número de agentes;
- Os valores mínimo e máximo da dimensão do vector que representa a opinião do agente;
- O tipo de limiar de afinidade (pode-se especificar um determinado valor ou então optar por percorrer todos os valores que variam entre 1 e a dimensão do vector que representa a opinião do agente);
- Valor da semente da sequência aleatória

Outros parâmetros caracterizam a forma como são executadas as simulações:

- Número de repetições a executar (foi implementada a possibilidade de correr repetições de uma simulação com os mesmos parâmetros de configuração);
- Dimensão dos quadrados que representam os bits dos vectores de opinião;
- Velocidade de execução do modelo e de actualização da matriz.

Em relação à apresentação dos resultados das simulações efectuadas, para além da representação gráfica e dinâmica das opiniões dos vários agentes, existem duas tabelas:

- Uma tabela que apresenta os vários grupos de opiniões existentes com os respectivos números de agentes (esta tabela é actualizada ao longo da simulação);
- Outra tabela apresentando o resultado de cada simulação mostrando os respectivos parâmetros principais de configuração (número de agentes, dimensão da opinião, limiar de afinidade), a repetição realizada, o número de ciclos realizados e o número de grupos obtidos (esta tabela é actualizada no fim da execução de cada simulação).

Em termos de controlo de execução do modelo, estão disponíveis três botões:

- Um botão que lança a execução de um ciclo de simulações;
- Outro botão que suspende a execução da simulação em curso;
- E, por fim, um botão que termina de imediato a execução.

Classes Específicas

A única classe específica que foi necessário definir neste modelo foi a classe “Population” derivada da respectiva classe definida na plataforma.

Foram adicionadas três propriedades:

- "AgentSize" – dimensão do vector opinião do agente;
- "CyclesNumber" – número máximo de ciclos a executar na simulação;
- "SimilarityThreshold" – limiar de afinidade.

Foi também criado um construtor específico para esta classe que complementa o construtor da classe base. Neste construtor, são criados os vários agentes do modelo com o vector que representa a respectiva opinião inicializado aleatoriamente, e são criados os vários grupos de agentes a considerar no início da simulação.

Componentes

Quatro componentes foram construídos para este modelo:

- "InitWorld";
- "SocializePopulation";
- "FinalizeWorld";
- "ShowWorld".

No componente "InitWorld", a principal acção é a criação da população do ambiente, contemplando as características fundamentais da simulação (número de agentes, dimensão da opinião e limiar de afinidade).

O componente "SocializePopulation" implementa as funcionalidades específicas deste modelo:

- Por cada agente da população, identifica aleatoriamente um agente para interagir com este;
- Identifica o número de bits de opinião diferentes entre os dois agentes;
- Se este número for inferior ao limiar de afinidade, altera aleatoriamente um dos bits diferentes no agente original;
- Após ter contemplado todos os agentes da população, verifica se os grupos actualmente formados ainda permitem alguma mudança de opinião futura; caso exacta mudança não seja possível, a simulação deverá terminar porque seguramente não haverão futuras mudanças de opinião em qualquer agente constituinte a população do modelo.

O componente "FinalizeWorld" só tem uma acção: actualização do ficheiro de log da simulação com o registo dos principais parâmetros de configuração e com os principais resultados da simulação:

- Índice da simulação;
- Índice da execução;
- Dimensão do vector opinião;
- Limiar de afinidade;
- Número de ciclos executados;
- Número de grupos de agentes.

O componente "ShowWorld" apresenta os resultados da execução de um ciclo da simulação, mostrando as opiniões actualizadas de todos os agentes, actualizando os vários que indicam os valores específicos ao ciclo de simulação (índice da execução, dimensão do vector opinião,

limiar de afinidade, índice do ciclo actual e número de grupos de agentes) e actualizando a tabela que regista os grupos que resultam ao longo dos vários ciclos de execução.

7.4. O Modelo de Segregação de Schelling

7.4.1. Enquadramento teórico

No início dos anos setenta, Thomas Schelling publicou um artigo (Schelling, 1971) sobre os vários factores que podem explicar segregação social e a sua evolução dinâmica, com o objectivo de explicar a segregação racial que, na altura, era possível encontrar em muitas cidades americanas apesar das leis anti-raciais existentes. Schelling apresenta neste artigo o que se poderá considerar como um dos primeiros exemplos de aplicação de um Modelo Baseados em Agentes ao estudo de um fenómeno social.

Dado que na época ainda não estavam disponíveis os meios informáticos que mais tarde se tornariam extremamente divulgados com o surgimento do computador pessoal e da microinformática, o modelo foi construído pelo próprio Schelling de forma manual recorrendo a um tabuleiro e peças (tipo jogo de damas) representando os dois tipos de agentes envolvidos.

O modelo é constituído por uma matriz quadrada cujas células são ocupadas por agentes que podem ter uma de duas cores (cor vermelha e cor verde). Em cada célula só se pode encontrar no máximo um agente e algumas células estão vazias. Em cada ciclo, cada agente observa os seus vizinhos (ou seja os agentes alojados nas 8 células adjacentes), determina a percentagem de vizinhos de cor diferente da sua e, se esta percentagem for superior a um limiar de tolerância (predefinido e igual para todos os agentes, independentemente da respectiva cor), o agente desloca-se para uma célula vazia em que a referida percentagem será inferior ou igual

ao limiar de tolerância. Obviamente esta recolocação do agente noutra célula poderá provocar a deslocação de outros agentes na medida em que pode criar situações em que outros agentes se sintam "infelizes" ou seja fiquem com uma percentagem de vizinhos com cor diferente superior ao limiar de tolerância.

Verifica-se que, a partir de uma população distribuída aleatoriamente e portanto de forma uniforme, e mesmo nos casos em que o limiar de tolerância é bastante alto, por exemplo igual ou acima de 50%, ou seja nos casos em que o agente sente-se bem, mesmo com vizinhos que são na maioria de cor diferente, são criadas, após vários ciclos, áreas de segregação com agentes da mesma cor. Isto é, mesmo quando os agentes aceitam viver com uma maioria de agentes vizinhos com cor diferente da sua, existe tendência para se criar segregação racial: facto claramente coincidente com o observado em muitos centros urbanos dos Estados Unidos no início dos anos setenta.

7.4.2. Implementação do modelo

Interface de Utilização

A interface de utilização do modelo é constituída por um ecrã em que se encontram definidas duas áreas principais (ver figura):

- Uma área para recolha dos parâmetros de configuração;
- Uma área com a matriz onde os agentes se encontram posicionados.

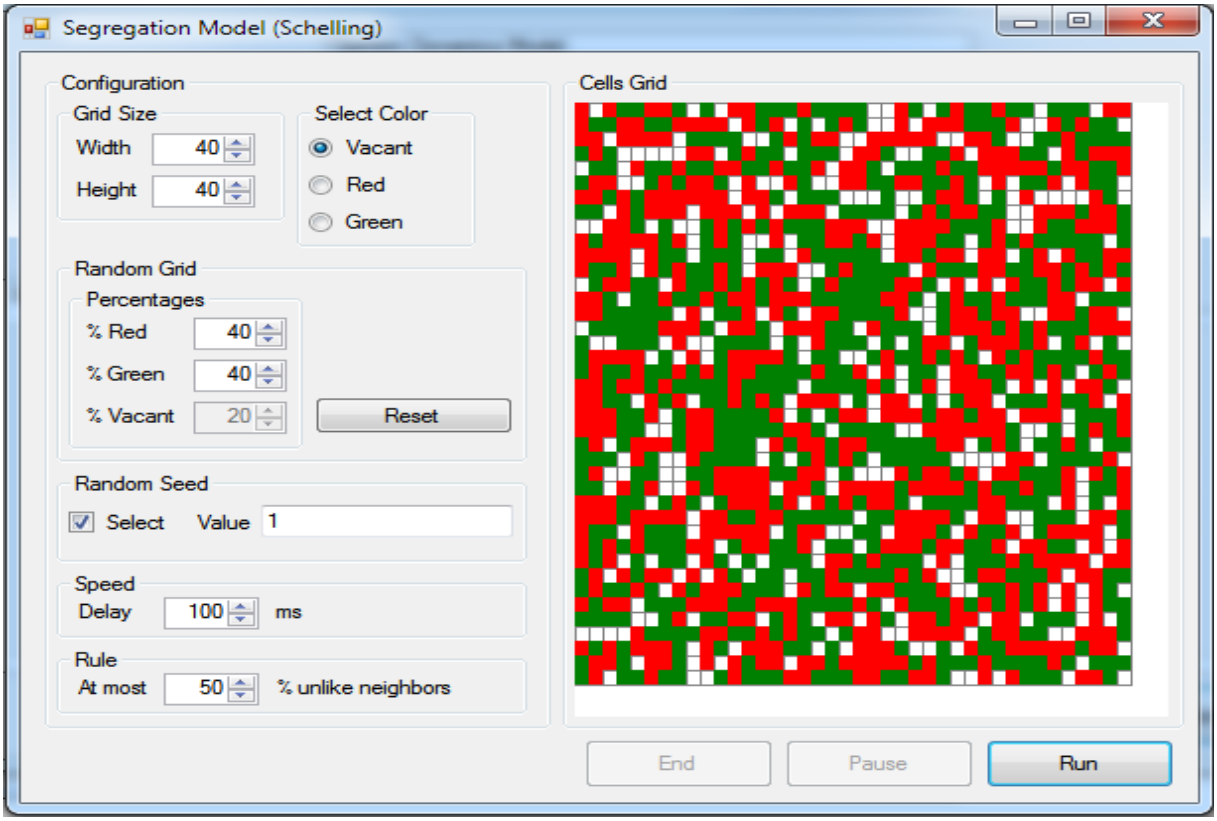

Figura 9 – O Modelo de Segregação de Schelling

A configuração do modelo é especificada através dos seguintes parâmetros:

- Dimensão da matriz em termos de largura e altura;
- Percentagens de agentes de cada cor;
- Limiar de tolerância dos agentes em relação aos vizinhos de cor diferente
- Semente da sequência de valores aleatórios;
- Velocidade de execução do modelo e de actualização da matriz.

A interface de utilização permite obter uma disposição aleatória inicial dos agentes cumprindo as percentagens indicadas de agentes de cada cor ou criar manualmente a configuração inicial.

Neste modelo, a apresentação dos resultados limita-se a mostrar graficamente a matriz de localizações dos agentes onde se podem verificar a existência, ou não, dos padrões de segregação.

Em termos de controlo de execução do modelo, estão disponíveis três botões:

- Um botão que lança a execução de um ciclo de simulações;
- Outro botão que suspende a execução da simulação em curso;
- E, por fim, um botão que termina de imediato a execução.

Classes Específicas

As únicas classes que foram adaptadas neste modelo foram a classe "Agent" e a classe "Population".

Na classe "Agent" foi adicionada a propriedade "Threshold" representando o limiar de tolerância dos agentes em relação aos vizinhos de cor diferente (comum a todos os agentes).

Na classe "Population", foi adicionado o método "GetUnsatisfiedAgentsList" que produz uma lista contendo todos os agentes que se devem considerar "não satisfeitos" por terem uma percentagem de vizinhos de cor diferente superior ao limiar de tolerância prefixado.

Componentes

Neste modelo foram criados três componentes:

- "SeekTarget";

- "FinalizeWorld";
- "ShowWorld".

No componente "SeekTarget", são identificados os agentes que não se encontram satisfeitos com a própria localização e são deslocados para novas localizações em que passam a ficar satisfeitos. Obviamente que se todos os agentes já estiverem satisfeitos com a própria localização, terá somente de ser assinalado que a simulação chegou ao fim de modo a que o motor da plataforma conclua a execução.

A funcionalidade do componente é implementada através de um único método privado ("MoveAgents"). Neste método é considerado cada agente que se encontra na lista de agentes "não satisfeitos" e é identificada a célula da matriz de localizações que satisfaça as seguintes condições:

- Não está ocupada;
- A percentagem de vizinhos de cor diferente não ultrapassa o limiar de tolerância prefixado;
- A distância à célula onde se encontra o agente é mínima.

Caso exista, o agente é movido para esta célula, sendo esta operação executada para todos os agentes que se encontram na lista de agentes "não satisfeitos".

O componente "ShowWorld" simplesmente actualiza a matriz de localização dos agentes, sendo executado em cada ciclo de execução da simulação.

Por fim, o componente "FinalizeWorld" prepara somente a interface de utilização, activando e inibindo os botões apropriados, considerando que a simulação concluiu.

8. Evolução da Plataforma Aplicacional

8.1. Análise crítica

Seguramente que uma crítica que se poderá fazer ao resultado do trabalho realizado é que a criação de um novo modelo com base na plataforma aplicacional desenvolvida exige que quem o faça tenha conhecimentos de programação, em geral, e conhecimentos do ambiente de desenvolvimento .Net. Esta consideração é claramente verdadeira e a crítica é certamente fundamentada: de facto esta exigência existe e, apesar da significativa evolução que a plataforma de desenvolvimento .Net tem tido em relação à facilidade de programação e à disponibilização de ferramentas de ajuda e documentação, são ainda indispensáveis conhecimentos específicos do ambiente de desenvolvimento .Net para modificar programas para este ambiente.

No entanto, por outro lado, também há que dizer que se escolheu desenvolver a plataforma aplicacional no ambiente .Net para contemplar as situações em que o investigador já possa ter familiaridade com este ambiente (é de facto actualmente um dos ambientes mais comuns para o desenvolvimento de aplicações) e não queira ter que adquirir os conhecimentos necessários para trabalhar com outras plataformas, eventualmente especificamente destinadas à criação de Modelos Baseados em Agentes (algumas destas plataformas foram referidas e sucintamente descritas num dos capítulos da presente dissertação).

Assim, na situação em que o utilizador já tenha experiência de programação e familiaridade com o ambiente .Net, recorrendo à plataforma aplicacional desenvolvida e tirando partido da arquitectura modular implementada, o utilizador poderá certamente criar um novo modelo de forma muito eficiente num curto prazo de tempo.

Uma outra crítica que poderá ser aplicada ao trabalho realizado é a inexistência de um configurador que permita definir algumas, senão todas, características de novos modelos a implementar; ou seja, não existe uma ferramenta que possibilitando adaptar alguns aspectos gerais da plataforma evite que o utilizados tenha que implementar, através de programação específica, essas adaptações.

De facto, a disponibilização de uma ferramenta deste tipo tem toda a razão de ser mesmo considerando que o autor tenha a capacidade de efectuar eficazmente, a codificação das adaptações necessárias: certamente, se essa ferramenta existisse, o esforço e o tempo necessários para a implementação das adaptações seriam menores.

Dada a dimensão do trabalho efectuado e dada a complexidade do desenvolvimento de uma ferramenta deste tipo com um alcance que contemple adequadamente os requisitos inerentes à criação de modelos muito variados, este desenvolvimento deverá ser considerado fora do âmbito da presente dissertação e fará parte das extensões e ampliações da plataforma.

8.2. Evolução e Ampliação

A partir do trabalho efectuado, e considerando algumas das necessidades principais que se colocam na aplicação de Modelos Baseados em Agentes ao estudo de fenómenos sociais, económicos e de outros tipos, podem-se identificar claramente algumas recomendações e sugestões para desenvolvimentos futuros.

A plataforma aplicacional desenvolvida e apresentada nesta dissertação já apresenta um âmbito e um nível de acabamento que permitem a sua utilização na construção de vários tipos de modelos.

Existem no entanto muitos aspectos a considerar no que respeita a evolução e a ampliação desta plataforma. Considera-se "evolução", a alteração de uma funcionalidade já implementada de modo a contemplar alguma característica ainda não disponível de um determinado tipo de modelo ou de uma funcionalidade transversal; considera-se "ampliação", a criação de uma nova funcionalidade, ou conjunto de funcionalidades, de modo a contemplar um novo tipo de modelos ou uma nova funcionalidade transversal.

Assim, no âmbito da evolução da plataforma podem-se considerar os seguintes tópicos:

- Versão Web;
- Versão Java;
- Padrões de desenho de SW;
- Documentação.

No âmbito da ampliação funcional da plataforma, podem-se considerar os seguintes tópicos:

- Captura e reposição de cenários;
- Análise estatística;
- Algoritmos genéticos;
- Redes de interacção;
- Configurador genérico.

8.2.1. Versão Web

Na sua versão actual, a plataforma aplicacional permite somente criar Modelos Baseados em Agentes que só podem ser executados no posto de trabalho do utilizador. Isto é, os modelos

obtidos são aplicações completas do tipo "Desktop Application" que correm no ambiente do sistema operativo do posto de trabalho do utilizador.

Uma evolução natural deste tipo de arquitectura aplicacional será transformá-la para uma arquitectura do tipo "Web Application". Com esta arquitectura, o modelo criado com a plataforma é executado num servidor Web e acedido pelo utilizador através do Internet Browser a partir do seu ponto de acesso à Internet (computador, telemóvel ou qualquer outros dispositivo que tenha acesso à Internet).

A principal vantagem da arquitectura "Web Application" é que deixa de ser necessário distribuir a aplicação que implementa o Modelo Baseado em Agentes para os postos de trabalho dos utilizadores: estes só precisam de ter acesso ao endereço Internet a partir do qual é possível executar o modelo.

8.2.2. Versão Java

Java é uma plataforma para desenvolvimento de aplicações alternativa à plataforma .Net da Microsoft, podendo-se afirmar seguramente que estas duas plataformas são as mais utilizadas actualmente para desenvolvimento de aplicações do tipo "Desktop" ou "Web".

Uma das vantagens que actualmente a plataforma Java apresenta, em comparação com a plataforma .Net, é que a primeira permite criar aplicações que podem ser executadas em vários sistemas operativos enquanto que as aplicações .Net correm praticamente só no sistema operativo Windows (apesar de várias incitativas em curso no sentido da generalização da plataforma .Net).

Assim, considerando esta vantagem, poderá ter sentido migrar a plataforma aplicacional desenvolvida para o mundo Java. Esta migração será certamente facilitada pelo facto que a

aplicação em questão foi criada numa linguagem orientada por objecto, linguagem C#, que apresenta muitas semelhanças com a linguagem Java.

8.2.3. Padrões de desenho de SW

No âmbito do desenvolvimento de *Software* em geral, e na programação orientada por objectos em particular, a utilização de padrões para a definição do melhor desenho a implementar tendo em vista a resolução de um determinado problema é uma estratégia fundamental para a realização de aplicações que sejam facilmente mantidas, tanto sob o ponto de vista da manutenção correctiva (correcção de erros) como do ponto de vista da manutenção evolutiva (alteração de funcionalidades existentes ou criação de novas funcionalidades).

Apesar da arquitectura da plataforma aplicacional aqui apresentada ter uma estrutura bem definida, tendo demonstrado eficácia e eficiência na construção de vários modelos, certamente será oportuno rever o desenho de alguns aspectos da plataforma à luz dos padrões de desenho actuais no sentido de tornar a plataforma mais robusta no que diz respeito à sua manutenção correctiva e evolutiva.

8.2.4. Documentação

Actualmente a documentação da plataforma desenvolvida é constituída pela presente dissertação. Dada a importância que a documentação de *Software* tem em relação à facilidade de manutenção desse *Software*, será importante reestruturar e completar a documentação actual de modo a obter uma documentação que seja de fácil consulta e que contemple a plataforma na sua totalidade e em todos os seus detalhes.

Esta revisão e aperfeiçoamento da documentação da plataforma também deverá ter em conta a necessidade de contemplar utilizadores de outras países e portanto será imprescindível produzir uma versão inglesa da documentação.

Por outro lado, haverá que considerar a utilização de ferramentas para a produção automática de documentação existentes no ambiente .Net ou outras ferramentas fornecidas por terceiros. A utilização destas ferramentas não só permite simplificar o processo de produção de documentação mas também sobretudo faz com que a estrutura da documentação produzida cumpra normas e padrões considerados de referência na produção de documentação.

8.2.5. Captura e reposição de cenários

A possibilidade de guardar, em qualquer momento, o estado completo de uma simulação para posterior continuação da sua execução é de grande importância sobretudo quando os tempos necessários para a execução dos modelos se tornam muito longos devido à complexidade dos modelos face aos recursos informáticos disponíveis.

Guardar o estado completo de uma simulação obriga a contemplar não só todas as variáveis que definem a simulação mas também todas os resultados já registados bem como os valores relevantes actuais necessários para repor a simulação exactamente no estado em que se encontrava quando foi suspensa.

8.2.6. Análise estatística

A análise estatística dos resultados, em muito Modelos Baseados em Agentes, é indispensável para a confirmação das hipóteses de investigação subjacentes às teorias formuladas para descrever os fenómenos estudados.

Deverá ser incluída na plataforma aplicacional, apresentada nesta dissertação, uma livraria de funções estatísticas que permitam utilizar os vários métodos disponibilizados pela Análise Estatística, de modo a que facilmente sejam incorporadas no modelo específico a avaliação estatística apropriada dos resultados.

Convém também referir como será recomendável, neste âmbito, incluir objectos que permitam apresentar as avaliações estatísticas de forma gráfica contemplando os vários tipos de diagramas utilizados neste tipo de análise.

8.2.7. Representação de redes

Frequentemente, na modelização baseada em agentes, é muito útil e até, por vezes, indispensável, analisar as redes resultantes das interacções realizadas entre os agentes constituintes o modelo, com o objectivo de identificar padrões que possam confirmar ou refutar as hipóteses teóricas assumidas.

Assim, uma plataforma que se pretenda adequada à criação de vários tipos de Modelos Baseados em Agentes deverá englobar instrumentos que permitam facilmente representar as redes em questão. Dada a complexidade que a representação destas redes envolve e dada a carga computacional que envolve, a escolha dos instrumentos adequados deverá ser feita considerando não só as funcionalidades pretendidas mas também as característica relevantes do ambientes tecnológicos em que a plataforma irá ser executada.

8.2.8. Algoritmos genéticos

Apesar de se ter visto previsto no ciclo principal de execução da plataforma os passos de selecção da população, cruzamento e mutação, fases típicas da implementação de algoritmos

genéticos, será muito útil criar um Modelo Baseado em Agentes que utilize um algoritmo deste tipo, com o fim de adaptar adequadamente a plataforma a modelos deste tipo.

Seguramente, e aliás como aconteceu na implementação dos modelos apresentados nesta dissertação, que a criação de um Modelo Baseado em Agentes que implemente um algoritmo genético irá comportar alterações nas classes fundamentais da plataforma e até, eventualmente, poderá obrigar a ajustar algum aspecto relacionado com a arquitectura da plataforma.

8.2.9. Configurador Genérico

De acordo com o referido no ponto sobre as críticas aplicáveis ao trabalho realizado, a disponibilização de um configurador genérico que permita definir alguns aspectos dos novos modelos a criar é uma adição interessante porque poderia contribuir significativamente á redução dos tempos envolvidos na construção de novos modelos.

9. Conclusão

9.1. Resposta à Questão Central de Investigação

A questão central de investigação assumida como orientadora ao presente trabalho foi a seguinte:

Quais as vantagens e quais as desvantagens de uma plataforma aplicacional genérica para o desenvolvimento de Modelos Baseados em Agentes?

Em vários pontos da dissertação são feitas observações e são tiradas conclusões que constituem uma parte da resposta a esta questão. No entanto convém concentrar neste ponto as seguintes observações e conclusões.

Vantagens

As principais vantagens a referir são:

- Caso o investigador possua familiaridade com programação em ambiente .Net, conseguirá, com facilidade e mais rapidamente, criar Modelos Baseados em Agentes utilizando a plataforma desenvolvida, não sendo obrigado a aprender outras linguagens ou paradigmas de programação;
- Dado que a plataforma foi desenhada com base num Quadro de Referência Conceptual, o investigador ao utilizar a plataforma será orientado na implementação dos conceitos fundamentais do desenvolvimento de Modelos Baseados em Agentes;

- A arquitectura modular da plataforma desenvolvida facilita a reutilização, num novo Modelo Baseado em Agentes, de componentes criados anteriormente para outros modelos;
- Como consequência da grande difusão do ambiente .Net, existem muitos componentes genéricos disponíveis na modalidade "Open Source", os quais poderão facilitar a criação de modelos com necessidades específicas sobretudo ao nível da interface de utilizador ou ao nível de algoritmos a implementar.

Desvantagens

Há que referir que algumas das desvantagens a seguir indicadas resultam do facto que a plataforma desenvolvida encontra-se na sua primeira versão e que, com a sua evolução conforme descrito anteriormente, alguns dos pontos negativos aqui referido deixarão de existir.

Destacam-se as seguintes desvantagens na utilização da plataforma:

- A criação de um novo modelo com base na plataforma aplicacional desenvolvida exige conhecimentos de programação, em geral, e conhecimentos específicos do ambiente de desenvolvimento .Net, para além da disponibilidade das ferramentas necessárias para programação .Net (geralmente é utilizado o Visual Studio da Microsoft)
- Em geral, o trabalho que o desenvolvimento de um modelo exige traduz-se em codificação, não existindo na versão actual da plataforma configuradores ou "*wizards*" que permitam efectuar uma parte deste trabalho sem recorrer à programação;

- A validação interna do modelo exige uma análise ao código implementado, o que pressupõe bons conhecimentos de programação, bem como um bom conhecimento da estrutura da plataforma;
- A execução do modelo criado com a plataforma exige que o executável da plataforma resida no posto de trabalho em que o modelo é executado, para além do ambiente .Net (este último requisito não é relevante porque o ambiente .Net para execução de programas está disponível gratuitamente);
- Na versão actual as funcionalidades relacionadas com Análise Estatística e representação de redes são inexistentes tendo de ser desenvolvidas nos casos em que sejam relevantes para o modelo específico a desenvolver.

Com base nestas considerações, conclui-se que a plataforma aplicacional desenvolvida, na sua versão actual, seja recomendável para as situações em que na equipa de investigação exista um elemento com conhecimentos de programação, em geral, e do ambiente de programação .Net, em particular. Nesse caso, tirando partido da plataforma, e eventualmente integrando-a com outros componentes disponíveis que contemplem alguma funcionalidade específica ainda não implementada ao nível da plataforma, poder-se-á desenvolver o modelo pretendido de forma muito eficiente. Em versões futuras da plataforma em que esta disponibilize algumas das extensões e ampliações referidas acima, os ganhos de eficiência serão muito mais significativos, e se a versão Java também se tornar disponível a exigência do ambiente .Net, e do seu conhecimento, deixará de ter sentido.

9.2. Recapitulação dos Objectivos e dos Resultados

O principal objectivo do presente Trabalho de Fim de Mestrado foi desenvolver uma plataforma aplicacional para o desenvolvimento de Modelos Baseados em Agentes.

Para isso, foi elaborado, com base em pesquisa bibliográfica, um Quadro de Referência Conceptual que orientasse a definição da arquitectura da plataforma. O ponto de partida para a construção deste quadro de referência foi a definição precisa dos conceitos e características subjacentes aos Modelos Baseados em Agentes.

Após o desenvolvimento da plataforma, esta foi utilizada para a criação de quatro modelos concretos, em domínios diferentes para o estudo de fenómenos concretos, que se podem considerar referências no âmbito da modelização nas Ciências Sociais. A aplicação da plataforma na construção destes quatro permitiu, por um lado verificar a sua adequabilidade e, por outro lado, afinar alguns aspectos que somente assim puderam ser identificados, nomeadamente a selecção de propriedades e métodos a implementar nas classes fundamentais da plataforma.

O terceiro objectivo do trabalho (identificação de vários melhoramentos e extensões à plataforma) encontra-se descrito no capítulo anterior.

Por fim, a resposta à questão central de investigação (o quarto e último objectivo do trabalho) foi descrita no presente capítulo.

Em complemento a estes objectivos, há que referir um objectivo geral e transversal a todo o trabalho realizado: adquirir e consolidar conhecimentos tanto ao nível da teoria subjacente aos Modelos Baseados em Agentes como ao nível do desenvolvimento de aplicações.

Os objectivos acima referidos podem-se considerar atingidos, tendo, como principal resultado visível e consistente, a disponibilização de um produto que pode ser realmente utilizado na criação de novos Modelos Baseados em Agentes.

Aliás, as fontes da plataforma desenvolvida também poderão ser disponibilizadas para que quem o deseje possa evoluir a plataforma no sentido de esta se tornar mais completa

funcionalmente, de modo a contemplar outros tipos de modelos ou funcionalidades mais completas mesmo para os modelos já contemplados.

Neste sentido, o candidato, considera que, após aprovação do Instituo Superior de Economia e Gestão, o código fonte possa ser disponibilizado à comunidade científica sob a forma de "Open Source", permitindo a partilha de todo o código fonte mas de uma forma controlada, de modo a que seja assegurada uma evolução coordenada do produto.

10. Bibliografia

Araújo, T., 2010. Introdução à Economia Computacional (em preparação).

Axelrod, R., 1997. The Dissemination of Culture: A Model with Local Convergence and Global Polarization. Journal of Conflict Resolution, Vol. 41, No. 2, 203-226.

Axelrod, R., 2005. Advancing the Art of Simulation in the Social Sciences. Gerald R. Ford School of Public Policy. University of Michigan.

Axelrod, R., Tesfatsion, L., 2005. A Guide for Newcomers to Agent-Based Modeling in the Social Sciences (anexo incluído no livro Handbook of Computational Economics, Vol. 2: Agent-Based Computational Economics, editado por Tesfatsion L. e Judd K., Elsevier B. V., 2006).

Axtell, R., 2000. Why Agents? On the Varied Motivations for Agent computing in the Social Sciences. Working Paper No. 17. Center on Social and Economic Dynamics. The Brookings Institution.

Banisch, S., Araújo, T., Louçã, J., 2009. Opinion Dynamics and Communication Networks. Working Paper No. 16. School of Economics and Management. Technical University of Lisbon.

Collier, N.T., North, M.J., Ozik, J., Tatara, E., 2007. Visual Agent-based Model Development with Repast Simphony. Em Proceedings of the Agent 2007 Conference on Complex Interaction and Social Emergence, Argonne National Laboratory.

Gilbert, N., 2004. Agent-based social simulation: dealing with complexity. The Centre for Research in Social Simulation - University of Surrey.

Gilbert, N., 2007. Agent-based models. The Centre for Research in Social Simulation - University of Surrey.

Gilbert, N., Terna, P., 2000. How to build and use agent-based models in social science. Mind & Society Journal, vol. 1, no. 1, 57-72.

Gilbert, N., Troitzsch, K. G., 2005. Simulation for the Social Scientist. Milton Keynes: Open University Press.

Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K., 2004. Mason: A new multi-agent simulation toolkit. Em: Proceedings of the 2004 Swarmfest Workshop.

Macal, C. M., North, M. J., 2005. Tutorial on Agent-Based Modeling and Simulation. Proceedings of the 2005 Winter Simulation Conference.

Macy, M. W., Willer, R., 2001. From Factors to Actors: Computational Sociology and Agent-Based Modeling. Annual Review of Sociology, vol. 28, 143-166.

McFadzean, D., 1994. SimBioSys: A Class Framework for Biological Simulations. Master of Science Thesis. Department of Computer Science, University of Calgary.

Minar, N., Burkhart, R., Langton, C., Askenazi M., 1996. The Swarm simulation system: A toolkit for building multi-agent simulations. Working Paper 96-06-042, Santa Fe Institute.

Railsback, S. F., Lytinen, S. L., Jackson, S. K., 2006. Agent-based simulation platforms: Review and development recommendations. Simulation, vol. 82, no. 9.

Schelling, T. C., 1971. Dynamic Models of Segregation. Journal of Mathematical Sociology, vol.1, 143-186.

Tesfatsion, L., 2002. Agent-Based Computational Economics: Modelling Economies as Complex Adaptive Systems. Department of Economics, Iowa State University.

Vriend, N., 2006. ACE Models of Endogenous Interactions (capítulo incluído no livro Handbook of Computational Economics, Vol. 2: Agent-Based Computational Economics, editado por Tesfatsion L. e Judd K., Elsevier B. V., 2006).

Whitehead, D., 2008. The El Farol Bar Problem Revisited: Reinforcement Learning in a Potential Game. University of Edinburgh.